The Delta Radiance Field

Vom Fachbereich Informatik
der Technischen Universität Darmstadt
genehmigte

DISSERTATION

zur Erlangung des akademischen Grades
Doktor-Ingenieur (Dr.-Ing.)

von
M.Sc. Tobias Alexander Franke
geb. in Frankfurt am Main

Referenten der Arbeit:	Prof. Dr. techn. Dieter W. Fellner Technische Universität Darmstadt
	Prof. Dr. techn. Michael Wimmer Technische Universität Wien
Tag der Einreichung:	29.7.2015
Tag der Disputation:	11.9.2015

D17
Darmstadt 2015

Zusammenfassung

Die weite Verbreitung von mobilen Endgeräten, welche in der Lage sind, realitätsnahe Bilder in Echtzeit zu berechnen, haben ein erneutes Interesse an der Forschung und Weiterentwicklung von Augmented Reality Anwendungen geweckt. Innerhalb des breiten Spektrums von vermischten reellen und virtuellen Elementen existiert ein spezieller Bereich mit dem Ziel, reale Szenen um virtuelle Kopien real existierender Objekte oder bald verfügbarer Produkte visuell plausibel zu erweitern. Überraschenderweise lässt allerdings der momentane Stand der Technik an genau dieser Stelle stark zu wünschen übrig: Augmentierende Objekte werden in aktuellen Systemen oft ohne jegliche Rekonstruktion von Umgebung und Beleuchtung in die reale Szene integriert und vermitteln den Eindruck, das Kamerabild einfach zu übermalen anstatt die Realität zu *erweitern*. Angesichts der Fortschritte in der Filmindustrie, die Vermischungen von Realitäten in allen Extremfällen bereits handhabt, ist es angebracht zu fragen, warum solche Erkentnisse nicht bereits ihren Weg zurück in den Augmented Reality Sektor gefunden haben.

Augmented Reality Anwendungen, welche grundsätzlich als Echtzeitandwendungen verstanden werden und die räumliche Zuordnung zwischen virtuellen und realen Elementen rekonstruieren, müssen zur Laufzeit auf weitere lückenhafte Informationen über die reale Szene reagieren. Darunter finden sich die unbekannten Beleuchtungsverhältnisse der realen Szene und die unbekannten Eigenschaften realer Oberflächen. Jedwede Rekonstruktion die zur Laufzeit ad-hoc geschieht muss mit einem entsprechenden Algorithmus arbeiten, der die Beleuchtung virtueller Objekte und den Transfer von virtuellem Licht auf echte Oberflächen ebenso ad-hoc berechnet. Der immersive Eindruck einer

Augmented Reality Simulation ist, abgesehen von Realismus und Genauigkeit des Darstellungsverfahrens, primär abhängig von ihrerer Reaktions- und Verabreitungsgeschwindigkeit. Alle Berechnungen die das Endbild betreffen müssen in Echtzeit durchgeführt werden. Diese Bedingung schließt jedoch viele Verfahren, die in der Filmproduktion zum Einsatz kommen, direkt aus.

Die verbleibenden Echtzeit-Optionen sind drei Problemen gegenübergestellt: Dem *Shading* von virtuellen Oberflächen unter Einbezug natürlicher Beleuchtung, der *Nachbeleuchtung* realer Oberflächen entsprechend der veränderten Beleuchtungssituation durch das neu eingefügten Objekt, und glaubhafte *globale Interaktion* von realem und virtuellem Licht. Diese Dissertation präsentiert neue Beiträge, um alle drei Probleme zu lösen.

Der aktuelle Stand der Technik baut auf sogenannten Differential Rendering Techniken auf, um Globale Beleuchtungsalgorithmen in AR Szenarien zu integrieren. Dieser einfache Ansatz hat jedoch einen rechenaufwändige Kehrseite, die die Möglichkeiten, glaubhaften globalen Lichttransfer zu simulieren, stark eingrenzt. Diese Dissertation untersucht neue Shading- und Nachbeleuchtungsalgorithmen, die auf einem neuen mathematischen Grundwerk aufbauen, welches Differential Rendering ersetzt. Die daraus resultierenden Algorithmen sind nicht nur effizienter als aktuelle, konkurrierende Verfahren, sondern erweitern das Feld um Effekte die bisher in keinen anderen Publikationen demonstriert wurden.

Abstract

The wide availability of mobile devices capable of computing high fidelity graphics in real-time has sparked a renewed interest in the development and research of Augmented Reality applications. Within the large spectrum of mixed real and virtual elements one specific area is dedicated to produce realistic augmentations with the aim of presenting virtual copies of real existing objects or soon to be produced products. Surprisingly though, the current state of this area leaves much to be desired: Augmenting objects in current systems are often presented without any reconstructed lighting whatsoever and therefore transfer an impression of being glued over a camera image rather than *augmenting* reality. In light of the advances in the movie industry, which has handled cases of mixed realities from one extreme end to another, it is a legitimate question to ask why such advances did not fully reflect onto Augmented Reality simulations as well.

Generally understood to be real-time applications which reconstruct the spatial relation of real world elements and virtual objects, Augmented Reality has to deal with several uncertainties. Among them, unknown illumination and real scene conditions are the most important. Any kind of reconstruction of real world properties in an ad-hoc manner must likewise be incorporated into an algorithm responsible for shading virtual objects and transferring virtual light to real surfaces in an ad-hoc fashion. The immersiveness of an Augmented Reality simulation is, next to its realism and accuracy, primarily dependent on its responsiveness. Any computation affecting the final image must be computed in real-time. This condition rules out many of the methods used for movie production.

The remaining real-time options face three problems: The *shading* of virtual surfaces under real natural illumination, the *relighting* of real surfaces according to the change in illumination due to the introduction of a new object into a scene, and the believable *global interaction* of real and virtual light. This dissertation presents contributions to answer the problems at hand.

Current state-of-the-art methods build on Differential Rendering techniques to fuse global illumination algorithms into AR environments. This simple approach has a computationally costly downside, which limits the options for believable light transfer even further. This dissertation explores new shading and relighting algorithms built on a mathematical foundation replacing Differential Rendering. The result not only presents a more efficient competitor to the current state-of-the-art in global illumination relighting, but also advances the field with the ability to simulate effects which have not been demonstrated by contemporary publications until now.

Acknowledgments

Rarely is a larger piece of work composed without any kind of critique, and in my case I'm glad to say that I have found peers who, beyond giving me feedback to my work, have helped me compose it. My special thanks is due to Peter Kán and Philipp Lensing with both of whom I've exchanged data and comments to each of our publications. Cheers!

I am grateful for the distinguished group of people considering this thesis and the support, discussions and in-depth knowledge in rendering and Augmented Reality necessary to judge its merit of my supervisor Dieter Fellner and second examiner Michael Wimmer, as well as the invaluable help of Arjan Kuijper for guiding me from its inception to its publication.

There is one person who perhaps is most relevant to this section for being a constant mood lifter, my fellow student, my colleague, and above all my friend Sebastian Wagner. We have shared one office for the longest time, and without derailing discussions from the daily business I probably would've peered across the edge of madness. Thank you for everything!

I want to extend my gratitude to both *The Internet Archive* and the *Lewis Walpole Library* for making available print artifacts which I have used at the beginning of each chapter. The Introduction image, an advertisement print for Philipsthal's Phantasmagoria show at the Lyceum, is courtesy of The Lewis Walpole Library, Yale University. All other chapter images where extracted from Marion Fulgence's *L'optique*, and Figure 1.1 from Étienne-Gaspard Roberts's *Mémoires*, both courtesy of The Internet Archive.

Contents

List of Figures

List of Tables

Chapter 1

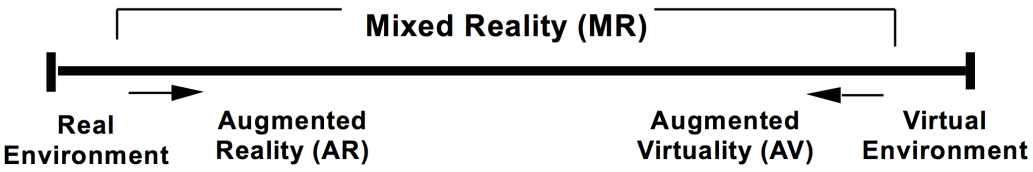
Introduction

In the midst of the 18th century a German coffee shop owner by the name of Johann Georg Schrepfer, who performed a series of stage shows with an appeal to the supernatural, convinced his audience that he could talk to the dead. To demonstrate this extraordinary claim, he superimposed pictures of the deceased onto smoke with the help of a projection device – the magic lantern – invented a century earlier. By immersing the imagery into the

stage environment he suggested his special gift to contact the afterlife to the audience [vK70, Mar69]. A Belgian contemporary expanded on this illusion with scaling, movement, and sound techniques to such degrees that attendees would often mistake trickery for reality. As audiences gradually lost the ability to make a clear distinction where the light show would end, authorities eventually stepped in to temporarily halt this new phenomenon called *Phantasmagoria* [Rob31].

Figure 1.1: *Robertson's Phantasmagoria in the Capuchin Crypt in 1797*: To the right, a man in the audience apparently aims a pistol at the apparition [Rob31].

Today, the pervasive use of computer-generated renditions of real world objects has largely blurred our own distinction between simulation and reality. Nowhere is this more apparent than in the movie industry: Where once special effects, scene props, or entire landscapes used to be crude scale replica, paintings, or other approximations to reality, the observer now finds himself unable to identify virtual copies of physical objects. Virtual previews have not only replaced their physical counterparts in movies, but also in advertising and other fields of pre-production. It is a method to visualize what would otherwise be either impractical, too time consuming, or prohibitively expensive.



Reality-Virtuality (RV) Continuum

Figure 1.2: The simplified reality-virtuality continuum as presented in [MTUK95]. One can expand the continuum into a plane, where in addition to the partitioning into real and virtual fractions the amount of realism from simple annotation to photorealism is projected.

The accumulated research in path-tracing over the last decade has shifted the focus from photorealism to solving complex sampling issues and efficient variance reduction. It is no longer a question *if* computer-generated images can appear real to a human observer, but rather a matter of computation and time constraints. While our methods for light transfer simulations still slowly converge back into the framework of our understanding of the physical nature of light, simulation and reality largely appear identical to most observers.

Between these two extremes, however, a continuum of *mixed realities* exists, a coherent fusion of two worlds into one common space. First introduced by Milgram et al. [MTUK95] as the *reality-virtuality continuum* (see Figure 1.2), classifications today take on different forms. In its original conception however, the continuum spans the range of all mixtures of real and virtual elements, where the fusion can vary in detail and realism, augmenting either reality with a wide bandwidth of additional pieces of information or transferring real objects into virtual space.

In the Augmented Reality section of this spectrum, reality is extended by introducing virtual elements into the scene. After geometric registration (i.e., reconstructing the position and orientation of the real camera with respect to the captured image) and scene reconstruction (i.e., identifying real light sources, real surfaces and their material properties in a captured environment), a proper algorithm can place and render an image. Depending on the

amount of a priori knowledge about the real scene, an algorithm can fuse one or more additional objects into the real scene which act coherently within their new context. Mutual shadowing, occlusion, lighting and other physical influences that would happen under normal circumstances, need to be computed to transport the impression to the observer that these new objects are really part of the real scene.

With the advent of computer-generated augmenting objects [NHIN86], today it is possible to build on recent advances in global light transport, tracking and high-fidelity reconstruction technology to create Mixed Reality images which appear plausible to the human observer. Path-traced solutions and properly reconstructed scenes using data-driven material measurements and 3D laser-scans of geometry deliver the methods necessary for this endeavor.

1.1 Problem Statement

Real-time augmentations of real image streams however cannot yet feasibly make use of these methods. If the user is to be convinced that the fused result is *real*, mere interactivity is as detrimental to the overall immersiveness as bad geometric registration. The following issues need to be addressed in order to solve the time-constraints in real-time Augmented Reality:

Problem 1: Shading of Augmenting Objects When shading augmenting objects, algorithms rely on proper reconstruction of the surrounding real space, which includes light sources, surfaces and material properties. Depending on the method to shade the object, these reconstruction processes can operate with varying degrees of freedom, which directly impacts the usability of the simulation in unknown environments. The algorithm should operate in real-time and also account for complex light interaction with various simulated materials.

Problem 2: Relighting of Real Surfaces Augmenting objects impact the appearance of their surrounding, blocking and scattering light from and into different directions. A real-time relighting algorithm has to account for these changes and properly add or subtract light on real surfaces, which need to be reconstructed live if the scene is unknown.

Problem 3: Global Illumination and Perception To create a solution for both Problem 1 and Problem 2 which can produce realistic and physically-based augmentations, the algorithms have to consider global light transport seamlessly changing from virtual to real space and vice versa. This includes the proper handling of light interacting with various types of materials such as metals to create the illusion of a fused reality.

1.2 Summary of Contributions

This dissertation explores and formulates new methods to augment a real camera image with a virtual object in real-time, shading and relighting the image in such way that the augmenting object matches real lighting conditions while at the same time ensuring that its effects on real light transfer are matched by the background image by adapting it for the change in illumination. In the subsequent chapters, the following contributions are presented:

Delta Radiance Field I propose a new view on relighting real environments with the formulation of the Delta Radiance Field. By deriving a linear transport operator to extract the difference between illumination conditions, I develop theory of light transfer between simulated augmenting objects and real surfaces. This new operator has performance benefits, which can be exploited in real-time global illumination relighting methods and addresses **Problem 3** while being the basis for a solution to **Problem 1** and **Problem 2**.

Image- and Volume-based shading of augmenting objects I explore image based lighting methods and come up with a set of two solutions to shade augmenting objects in unknown illumination conditions. These solutions can however impact flexibility or runtime behavior and are therefore suited for different situations, for instance when assuming rigidity in augmenting or real objects. Additionally, I present two new volumetric global illumination solutions to shade augmenting surfaces according to extracted point lights from the real environment. Whereas image based algorithms behave more robust under complex and rapidly changing lighting conditions, they cannot account for local lighting. To support local lights, I inject indirect light bounces from the reconstructed real surrounding of an object and from itself into a volume, clustering many indirect bounces into a scalable container to simulate transfer from real and virtual surfaces onto virtual ones in real-time. A combination of all these methods is derived to solve **Problem 1**.

Volume-based relighting of reality Based on the Delta Radiance Field formulation, I explore methods to simulate light transfer in small volumes around an augmenting object. After an extraction of the light differential on the operator level, both direct and indirect light is captured inside a voxelization of the scene. Three methods are proposed which represent and simulate light differently: Through precomputation of transfer, by diffusion propagation and by a pre-filtered gathering scheme to simulate transfer of real and virtual light from and to surfaces of varying roughness. In a final grand comparison between these methods, the state-of-the-art and ground truth results, strengths and weaknesses are exposed to determine a solution for **Problem 2**.

1.3 Publications

Key parts of this dissertation were already published in conference proceedings. The following publications are directly relevant and were incorporated with minimal editing.

[**Fra14a**] Tobias Alexander Franke. Delta voxel cone tracing. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, pages 39–44, Sept 2014

[**Fra13a**] Tobias Alexander Franke. Delta light propagation volumes for mixed reality. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 125–132, Oct 2013

[**FKOJ11**] Tobias Alexander Franke, Svenja Kahn, Manuel Olbrich, and Yvonne Jung. Enhancing realism of mixed reality applications through real-time depth-imaging devices in x3d. In *Proceedings of the 16th International Conference on 3D Web Technology*, Web3D '11, pages 71–79, New York, NY, USA, 2011. ACM

[**FJ08a**] Tobias Alexander Franke and Yvonne Jung. Precomputed radiance transfer for x3d based mixed reality applications. In *Proceedings of the 13th international symposium on 3D web technology*, Web3D '08, pages 7–10, New York, NY, USA, 2008. ACM

[**FJ08b**] Tobias Alexander Franke and Yvonne Jung. Real-time mixed reality with gpu techniques. In *GRAPP 2008: Proceedings of the Third International Conference on Computer Vision Theory and Applications*, pages 249–252. INSTICC Press, 2008

1.4 Outline

The remainder of this dissertation is organized as follows.

Chapter 2 introduces fundamental basics of light transport and related work on real-time global illumination algorithms, Augmented and Mixed Reality setups, tracking and reconstruction of real world physical objects.

Chapter 3 derives the Delta Radiance Field, which is the framework this dissertation builds on to explain the change in illumination when modifying scat-

tering events in a scene. This framework is ultimately used to find new, more efficient real-time global illumination relighting algorithms.

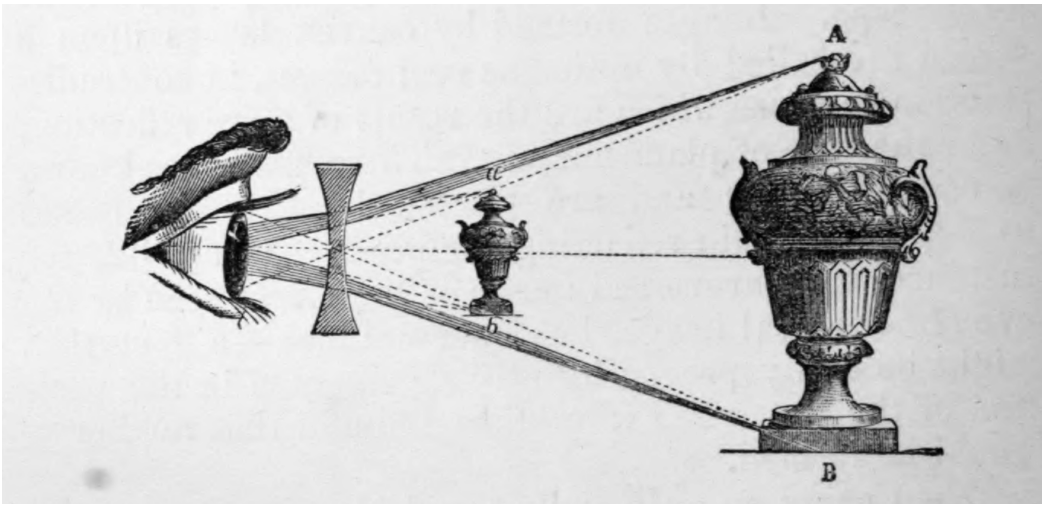
In Chapter 4 I elaborate on the subject of shading augmenting objects coherently within the frame of the real environment they are exposed in. Depending on certain assumptions about the virtual scene such as rigidity, precomputation and the use of measured materials is possible. I therefore present two methods for two different types of virtual objects, dynamic and static, working with real natural illumination.

Chapter 5 presents three successive solutions to the real-time relighting problem of real surfaces and related publications. These solutions are directly derived from the Delta Radiance Field equation of Chapter 3 to create a flexible, yet efficient global illumination algorithm which supports inter-transfer of radiance between virtual and reconstructed real objects.

Finally Chapter 6 concludes this dissertation with a summary and a section on open issues in real-time Augmented Reality.

Chapter 2

Fundamentals



2.1 Light Transport

2.1.1 Geometrical Optics

Light is a form of energy carried as electromagnetic radiation. Its behavior displays characteristics of two different theories: Wave theory and particle

$\langle\rangle_+$	Dot product clamped to positive numbers
$\mathbf{x}$	A surface point
λ	Wavelength
$\vec{n}$	Surface normal vector
$\vec{m}$	Microfacet normal
$\vec{h}$	Half-vector between a surface normal $\vec{n}$ and another vector $\vec{v}$
D	Microfacet/Normal Distribution Function
F	Fresnel function
G	Bidirectional geometric shadowing-masking term
f_r	Bidirectional Reflection Distribution Function
f	Bidirectional Scattering Distribution Function
f_d	The diffuse part of a BRDF
f_s	The specular part of a BRDF
α	Surface roughness parameter
ρ_d	Diffuse reflectance
ρ_s	Specular reflectance
Ω	Hemisphere above a point $\mathbf{x}$
$\vec{\omega_o}$	Exit direction of light
$\vec{\omega_i}$	Direction to incident light source
$\overline{\Omega}$	The path-space
$\overline{x}$	A path of a particle
$f_j(\overline{x})$	Measurement contribution function
ξ	Uniform random numbers $\in [0, 1)$
$\mathbf{T}, T_{ij}$	Linear transport operator and its coefficients
$\dot{t}, t_c$	Coefficient vector and its coefficients

Table 2.1: Summary of symbols and notations.

theory. Wave theory suggests that light spreads much like a water wave, with its frequency perpendicular along its direction of propagation. Different waves can interfere with each other causing effects such as polarization or diffraction. The famous double slit experiment by Thomas Young was used to show this property. In a certain band of wavelengths λ, light can be experienced by the human eye (see Figure 2.1).

In an attempt to explain black body radiation, Max Planck suggested that these waves however are packets of energy which can change only in discrete amounts instead continuously. While he called them *quanta* (Latin for *how much*), they eventually became known as *photons* as named by Gilbert Newton Lewis in 1926. The unified theory of light today, which regards light as neither wave nor particle but as phenomenon with properties of both, is

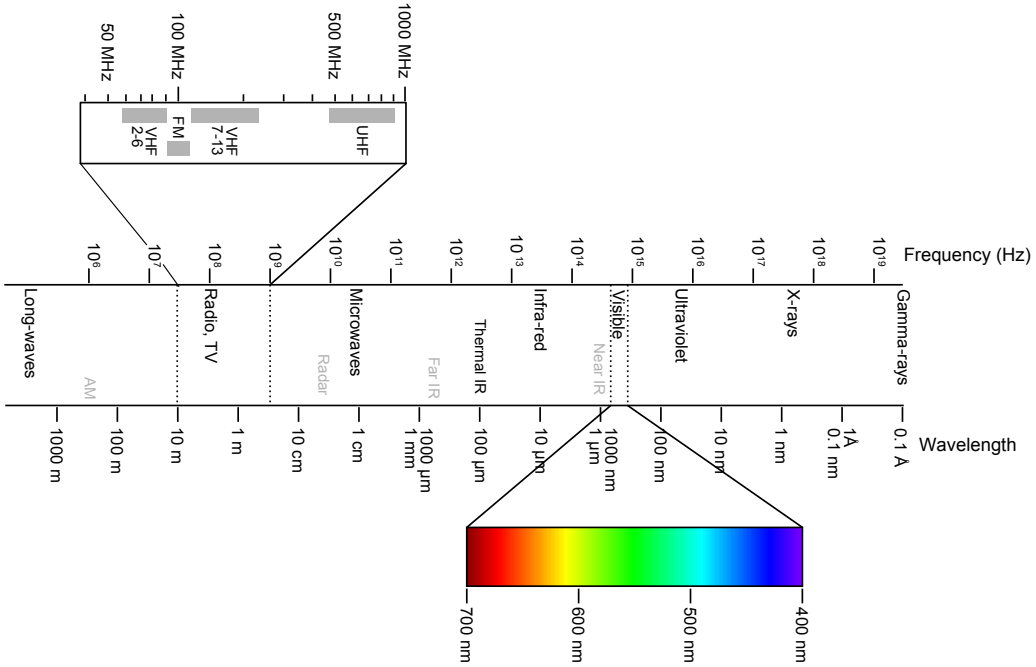


Figure 2.1: The electromagnetic spectrum. Wavelengths λ in the region between $370nm$ and $730nm$ composes the visible spectrum of the human eye. Image courtesy of Victor Blacus [Bla12].

called quantum mechanics.

When light is seen from scales much larger than its wavelength, its observed behavior is much simpler. Light on such scales travels in straight lines and is best described with simple laws governing reflection and refraction, all of which can be derived by *Fermat's principle* (also called *principle of least time*), which states that a path between two points taken by a ray of light is the path that can be traveled in the least time. This view of light is called *geometrical optics* and is usually used in computer graphics as a straightforward and pragmatic abstraction. While it cannot account for effects of light explained by higher-level models – diffraction and interference (wave optics), polarization and dispersion (electromagnetic optics), fluorescence and phosphorescence (quantum optics) – it simplifies the mathematical framework drastically.

If not stated otherwise, this work builds on the geometrical optics abstraction. Even within the boundaries of this limited framework all perceptually impor-

tant effects for this work can be simulated. In Table 2.1 symbols relevant to this thesis are listed in an overview.

2.1.2 Radiometry

The study of the propagation of electromagnetic radiation is called *radiometry* and is measured in wavelengths λ. It is not to be confused with *photometry*, which is the study of light as perceived brightness by the human eye. Of particular interest is the region of wavelengths λ between $370nm$ and $730nm$, because these correspond to the light which is visible to the human eye. In this section, I will review basic quantities and formulas to express these measurements.

Radiant energy Energy carried by an electromagnetic wave (or photons) is called *radiant energy*, denoted as Q and measured in joules (J). In particular, it is the amount of energy emitted by a light source over a period of time.

Radiant flux *Radiant flux*, also referred to as *radiant power* or simply *power*, is the total amount of energy passing through a surface per unit time t, measured in joules per second ($\frac{J}{s}$) or watts (W). It is denoted by the symbol Φ.

$$\Phi = \frac{\mathrm{d}Q}{\mathrm{d}t} \tag{2.1}$$

A light source's total emission is usually described with the term flux.

Irradiance and Radiant Exitance *Irradiance* is the area density of flux arriving on a surface A. It is denoted as E and measured in watts per square-meter ($\frac{W}{m^2}$).

$$E = \frac{\mathrm{d}\Phi}{\mathrm{d}A} \tag{2.2}$$

The incident power Φ is usually restricted to the upper hemisphere above the surface. In case of a projection of the area $\mathrm{d}A$ has to be weighted by a cosine response $\cos\theta$.

Radiant Exitance, called M, or *Radiosity*, referred to as B, is power leaving per unit surface area. The formula is identical to the irradiance Equation (2.2) with Φ referring to exit power over one hemisphere instead of incident power. For this reason, irradiance is also sometimes referred to as flux leaving an area.

Intensity In order to define *intensity*, it is first necessary to introduce the notion of a *solid angle*. The solid angle is the total area s subtended by an object when projected onto the a unit sphere. The entire sphere subtends a solid angle of 4π and respectively 2π for the hemisphere. Solid angles are measured in *steradians*.

Intensity is defined as flux density per solid angle $\mathrm{d}\omega$.

$$I = \frac{\mathrm{d}\Phi}{\mathrm{d}\omega} \tag{2.3}$$

Intensity is related to Irradiance: The solid angle $\mathrm{d}\omega$ can also be defined as $\frac{\mathrm{d}A}{r^2}$ where r is the distance to the emitter. By substituting the solid angle in Equation (2.3), it is easy to see the relation to E:

$$\frac{I}{r^2} = \frac{\mathrm{d}\Phi}{\mathrm{d}A} = E \tag{2.4}$$

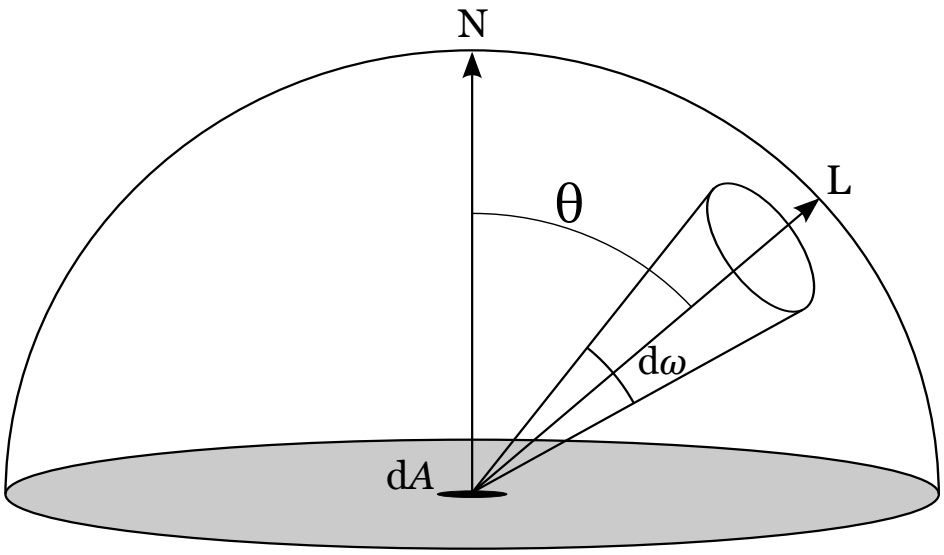


Figure 2.2: Radiance is flux per projected unit area ($\mathrm{d}A \cdot \cos\theta$) per unit solid angle ($\mathrm{d}\omega$).

It also follows from Equation (2.4) that irradiance has an inverse-square falloff with the distance r.

Radiance Perhaps the most frequently used and important radiometric quantity is *radiance*, which is the flux density per unit projected area per unit solid angle. The SI unit of radiance is watts per steradian per square-meter ($\frac{W}{\mathrm{sr}\cdot\mathrm{m}^2}$).

$$L = \frac{\mathrm{d}^2\Phi}{\mathrm{d}\omega \mathrm{d}A \cdot \cos\theta} \tag{2.5}$$

For a better visual representation, see Figure 2.2. Radiance has two important properties: First, all other radiometric quantities can be derived from given radiance by computing the integral of radiance over an area and directions. Second, it remains invariant along a ray through empty space.

$$L(\mathbf{x}, \vec{\omega}) = L(\mathbf{x} + t\vec{\omega}, \vec{\omega}), t > 0 \tag{2.6}$$

Radiance is often distinguished with a qualifier to clarify the direction of radiance in a given context. *Incident radiance* (i.e., photons arriving at some point $\mathbf{x}$ from direction $\vec{\omega}$) is usually denoted as $L_i(\mathbf{x}, \vec{\omega})$, whereas *exitant radiance* (i.e., photons leaving from some point $\mathbf{x}$ in direction $\vec{\omega}$) is denoted as $L_o(\mathbf{x}, \vec{\omega})$ or simply $L(\mathbf{x}, \vec{\omega})$.

For a more thorough discussion of radiance I refer the reader to the dissertation of Eric Veach [Vea98].

2.1.3 Interaction of Light and Matter

A generic surface reflection framework used in this thesis is presented by James T. Kajiya [Kaj86]:

$$L\left(\mathbf{x}, \vec{\omega_o}\right) = L_e\left(\mathbf{x}, \vec{\omega_o}\right) + \int_\Omega f\left(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}\right) L_i\left(\mathbf{x}, -\vec{\omega_i}\right) \cos\theta_i d\vec{\omega_i} \tag{2.7}$$

For each point $\mathbf{x}$ on a surface, the accumulated reflected radiance into a given exit direction $\vec{\omega_o}$ (also referred to as $\vec{V}$ in graphics literature) is the self-emission of the surface L_e plus the integral over all incident light directions $\vec{\omega_i}$ of the hemisphere Ω above $\mathbf{x}$ of its material response f, which handles the transfer from incident radiance L_i from direction $-\vec{\omega_i}$ to $\vec{\omega_o}$ with the Lambertian cosine emission law scaling it according to the surface normal $\vec{n}_\mathbf{x}$, where $\cos\theta_i = \langle \vec{n_\mathbf{x}}, \vec{\omega_i} \rangle_+$. Because this thesis deals with light in terms of geometrical optics, the wavelength dependency λ is dropped from the original equation.

When light strikes matter, it is *transferred* by the interaction with the material of the surface. The effect of the material on light is defined by a property

called the *Index of Refraction* (IOR) which is a complex number[1]. The real part indicates the effect on the speed of light (i.e., the amount it is slowed down compared to the speed in vacuum c), whereas the imaginary part determines whether it is absorbed or scattered by the material. Absorbed light is usually converted to another form of energy such as heat, which is disregarded in most computer graphics implementations.

The function which represents light transfer through the material, called Bidirectional Scattering Distribution Function (BSDF), is most easily expressed by the ratio of incident radiance L from direction $\vec{\omega_r}$ and the irradiance E into direction $\vec{\omega_i}$ per solid angle per unit projected area.

$$f\left(\mathbf{x}, \vec{\omega_i}, \vec{\omega_r}\right) = \frac{\mathrm{d}L(\vec{\omega_r})}{\mathrm{d}E(\vec{\omega_i})} \tag{2.8}$$

Scattered light is either *reflected* or *refracted* when hitting a surface. Reflected light may scatter in different directions, depending on the surface. Refracted light scatters beneath the surface one or multiple times before exiting possibly at a different position. This behavior is called transmission and can be seen in half-translucent materials such as marble or thin leaves.

Light with an angle of incidence θ_i is reflected in direction θ_r where the relationship of these angles is given by the *Law of Reflection*:

$$\theta_i = \theta_r \tag{2.9}$$

When Light moves from a medium with an IOR n_1 to a medium with an IOR n_2, its angle of refraction θ_t is given by *Snell's Law*:

[1] The IOR varies with wavelength, an effect which can be seen in the chromatic dispersion of light when passing through a prism.

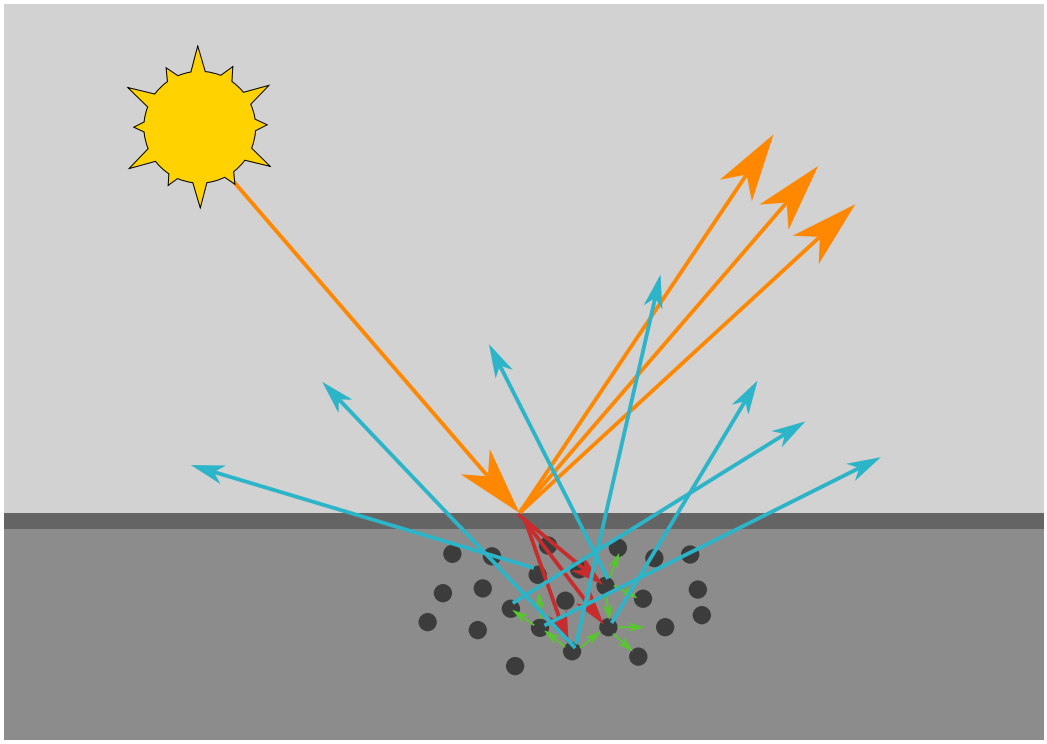


Figure 2.3: BSDF scattering behavior: Diffuse scattering can be simplified to a Lambert equation *if* the sampled area is larger than the scattering distance within the material. In this case, material interaction can be evaluated locally.

$$\frac{\sin(\theta_i)}{\sin(\theta_t)} = \frac{n_2}{n_1} \tag{2.10}$$

Light which roughly exits into the direction it entered from – slightly spreading the area it is reflected from – is a diffusion process responsible for the matte appearance of many non-conducting materials such as cloth. This process is depicted in Figure 2.3. When the sampled area is larger than the area produced by inscattered diffuse light, and when additionally transmission *through* the surface is ignored (i.e., a property of translucent materials), a BSDF equation can be simplified into a Bidirectional Reflectance Distribution Function (BRDF), which handles light-matter interaction locally. This function operates under the assumption that diffuse reflection ρ spreads incident light into all directions of the hemisphere of the local area equally.

Local diffuse operations can be expressed with the Lambertian reflectance equation.

$$
\begin{aligned}
L\left(\mathbf{x}, \vec{\omega_o}\right) &= \int_{\Omega} \rho_d \langle \vec{n}, \vec{\omega_i} \rangle_+ d\vec{\omega_i} && (2.11)\\
&= \rho_d \int_{\Omega} \cos\theta_i d\omega_i && (2.12)\\
&= \rho_d \int_0^{\pi} \int_0^{2\pi} \cos\theta_i \sin\theta_i d\Theta_i \Phi_i && (2.13)\\
&= \pi\rho_d && (2.14)
\end{aligned}
$$

From this result we can derive an ideal diffuse reflectance function f_d with constant diffuse reflectance ρ_d and an energy normalization coefficient $\frac{1}{\pi}$:

$$f_d(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}) = \frac{\rho_d}{\pi} \tag{2.15}$$

The main benefit of a BRDF is that the local expression of light reflection is independent of other surface points and can therefore be computed for each point in parallel.

Material functions may also be described as non-analytical, data-driven models [MPBM03]. The general framework computes light transport with the help of a special base with which measured material data is indexed.

2.1.3.1 Physically Based Rendering

Synthesizing images can follow different goals. For instance, the image could be generated with the aim to create an artistically pleasing result. In real-time rendering processes, many effects are often achieved with independent processes which have to adhere to a common set of rules. Physically Based Rendering (PBR) aims to create images with material and light definitions which closely relate to physical properties rather than aiming at visually

pleasing results. It is therefore necessary to have material definitions which do not violate physical concepts. The following are conditions a physically based BSDF has to adhere to.

Positivity The value of the BSDF is always positive.

$$f\left(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}\right) \geq 0 \tag{2.16}$$

Energy Conservation The total amount of energy reflected over all directions of the surface must be less or equal to the total amount of energy incident to it. In practical terms this means that the visible energy (i.e., reflected light) can at best decrease after bouncing off a surface, while the rest turns to heat or some other form which is not part of the simulation. A non-emissive surface however cannot emit more light than it received.

$$M = \int_{\Omega} L\left(\mathbf{x}, \vec{\omega_o}\right) \cos\theta_o d\vec{\omega_o} \leq \int_{\Omega} L\left(\mathbf{x}, \vec{\omega_i}\right) \cos\theta_i d\vec{\omega_i} = E \tag{2.17}$$

$$\forall \vec{\omega_o}, \int_{\Omega} f_r(\mathbf{x}, \vec{\omega_o}, \vec{\omega_i}) \cos\theta_i d\vec{\omega_i} \leq 1 \tag{2.18}$$

Helmholtz Reciprocity The standard assumption in geometric optics is that exchanging in- and outgoing light direction $\vec{\omega_i}$ and $\vec{\omega_o}$ in the BSDF does not change the outcome.

$$f\left(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}\right) = f\left(\mathbf{x}, \vec{\omega_o}, \vec{\omega_i}\right) \tag{2.19}$$

It is clear from this definition that some of the classical models used in

computer graphics such as the standard specular Phong model in the now deprecated OpenGL fixed function pipeline do not model physically plausible behavior. Blinn-Phong for instance is not energy-conserving and will loose brightness with increasing specularity. In some cases, this type of error can be addressed with a scaling mechanism to combat wrong energy output. Moreover, physically based models need to account for effects such as Fresnel and should exhibit parameters which have physically plausible meaning.

A new theory of specular reflection is therefore necessary to address the shortcomings of older models.

2.1.3.2 Microfacet Theory

To model specular reflectivity, physically based BRDF models are typically built on the theory of *Microfacets*. Microfacet theory suggests that the surface of an object exhibits a certain type of irregularity when viewed at microscopic detail. A surface which appears flat is actually composed out of many tiny, perfectly specular mirrors, i.e., the surface has variation which is smaller than the scale of observation. These tiny surface areas have a configuration which is said to form a *smooth* or *rough* macro scale.

The landscape these microscopic mirrors build is responsible for certain effects: Parallel rays which on a macro scale appear to be a single coherent ray of light are possibly reflected into different directions, blurring the appearance of the reflected light. A smooth surface forms a smooth reflection (i.e., perfectly specular), rough surfaces a rough one (i.e., varying degrees of glossiness). Furthermore, rough surfaces can cause slight amounts of self-shadowing and interreflection.

In Figure 2.4 an example is shown: The appearance of the macroscopic surfaces is caused by the rough configuration of the microfacets orientation on the microscopic scale.

For rendering applications however, dividing up any type of surface into tiny microfacets is unfeasible with regards to computation and memory require-

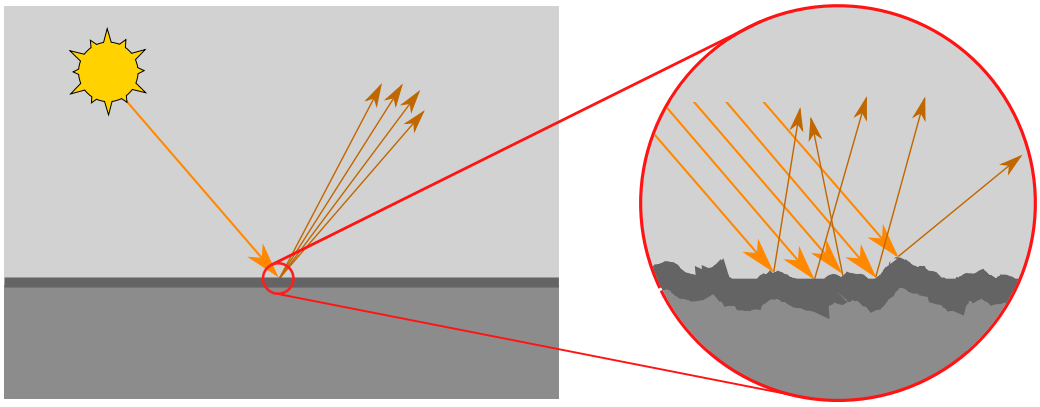


Figure 2.4: When magnifying an optically flat surface which exhibits glossy reflections, a landscape of microfacets appears. These microfacets scatter incident light into slightly different directions, giving the reflection its blurry characteristic. Rough surfaces also exhibit bounces between microfacets as well as self-shadowing behavior.

ments. Microfacet models therefore represent the configuration of a type of surface with a roughness α by statistical means: The overall self-shadowing, scattering between microfacets and blurring of incident light depends on a set of fixed parameters such as α.

The framework of a microfacet BRDF which handles the specular part f_s is known as the Torrance-Sparrow or Cook-Torrance model [TS67, CT82] [2], where $\vec{n}$ is the surface normal and $\vec{h} = \frac{\vec{\omega_o}+\vec{\omega_i}}{|\vec{\omega_o}+\vec{\omega_i}|}$ is the half-vector of $\vec{\omega_i}$ and $\vec{\omega_o}$:

$$f_s\left(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}\right) = \frac{F\left(\vec{\omega_i}, \vec{h}\right) G\left(\vec{\omega_i}, \vec{\omega_o}, \vec{h}\right) D\left(\vec{h}\right)}{\pi \langle \vec{n}, \vec{\omega_i} \rangle_+ \langle \vec{n}, \vec{\omega_o} \rangle_+} \tag{2.20}$$

This model includes the following components:

The Fresnel term F Fresnel is a function to determine the reflectance of smooth surfaces using only the refractive index and the angle of incidence. This value typically stays constant for the first 45 degrees of incidence and

[2] More recent publications use 4 instead of π as normalization factor for f_s [WMLT07].

can be thought of as specular color F_0 which represents the characteristic appearance at 0° incidence.

The geometric term G This term models the self-shadowing behavior of the microfacets on the surface and can be thought of as a visibility factor for a micro-landscape which simply depends on one parameter for the surface roughness.

The Normal Distribution Function D The NDF is a scalar term which determines the distribution of microfacet normals $\vec{m}$ oriented into a given direction. If more microfacets are oriented in the half-vector direction $\vec{h}$, the specular highlight will be brighter. Since D is a *density probability* of normals oriented in direction $\vec{h}$, its range is not restricted to $[0, 1]$: A high value of D indicates a high concentration of microfacets $\vec{m} = \vec{h}$.

The function D determines the overall brightness, shape and size of the specular highlight. Several propositions for Normal Distribution Functions exist in graphics literature: Cook-Torrance [CT82], Oren-Nayar [ON94], Beckmann [BS87], Schlick [Sch94], Ward [War92], Trowbridge-Reitz (also known as GGX) [TR75, WMLT07].

The complete BRDF is defined as follows:

$$f_r\left(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}\right) = f_d\left(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}\right) + f_s\left(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}\right) \tag{2.21}$$

The individual terms F, G and D can be configured with different analytical or data-driven functions when one model or another better represents a specific class of materials. For instance, an NDF specific for cloths may not be able to simulate other types of materials. However, this does not affect the Fresnel part of the BRDF, which can be left intact with its own approximation. A popular and computationally cheap variant for F is Schlick's approximation [Sch94]:

$$R(\theta) = R_0 + (1 - R_0)\left(1 - \left\langle \vec{\omega_o}, \vec{h} \right\rangle_+\right)^5 \tag{2.22}$$

$$R_0 = \left(\frac{n_1 - n_2}{n_1 + n_2}\right)^2 \tag{2.23}$$

R_0 is the *reflection coefficient* at the interface of the surface for light incident parallel to the normal, n_1 is the IOR of the medium from which light changes into another medium with IOR n_2.

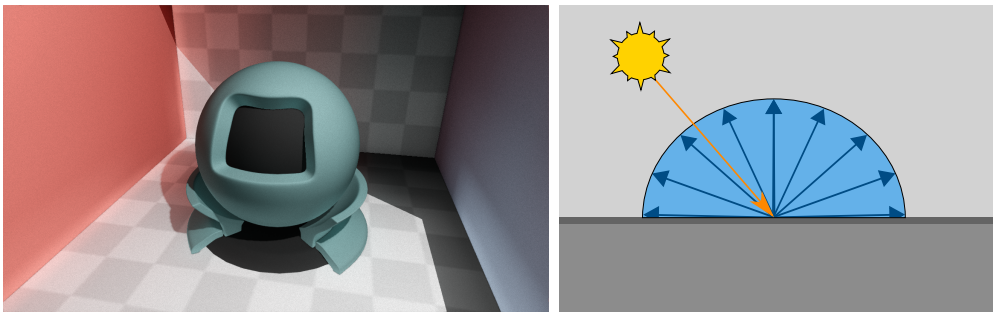
In Figure 2.5 an overview of several material interactions can be seen: A smooth diffuse composition of two matte rubber types in 2.5(a), a rough dielectric (i.e., a surface with diffusion components and a specular coating) in 2.5(b), a rough golden metal surface in 2.5(c) as well as a polished golden surface in 2.5(d).

2.1.4 Global Illumination

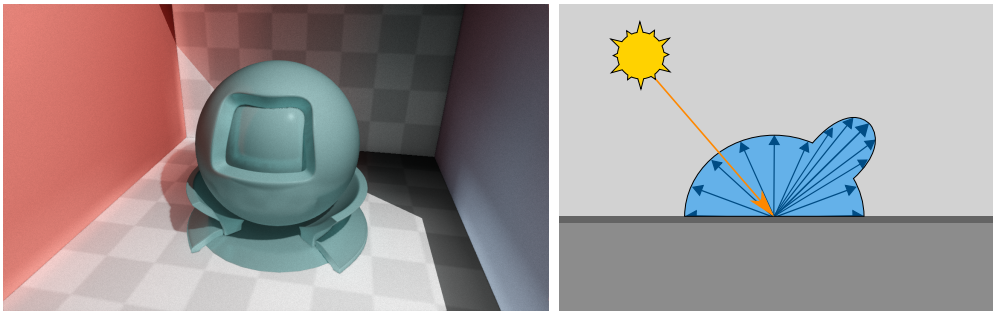
Interaction of light and matter may not stop the propagation of energy directly. Light which is not fully absorbed is still present in the scene and *bounces* into different directions, further interacting with other surfaces. This recursive behavior can be seen in the Rendering Equation (2.7): Incident light L_i is simply the integrated radiance from some other point in space $\mathbf{x}$. Algorithms which take into account this kind of indirect interaction in light transport equations are called *global*, computing how light bounces off one surface to interact with another.

2.1.4.1 The Path Integral Formulation

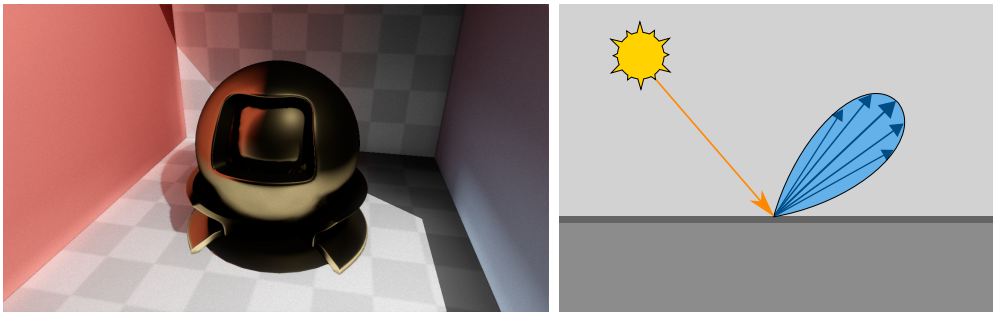
To better understand the problem of Global Illumination (GI), I would like to define the notion of a path first. In this form, the route or trajectory a particle takes from an emitter to a receiver through a scene is called a *light transport path* $\overline{x} = x_0, x_1 ..., x_n$ with n vertices x_k, where the direction of an edge between two vertices is written as $x_k \rightarrow x_{k+1}$. In an environment of opaque



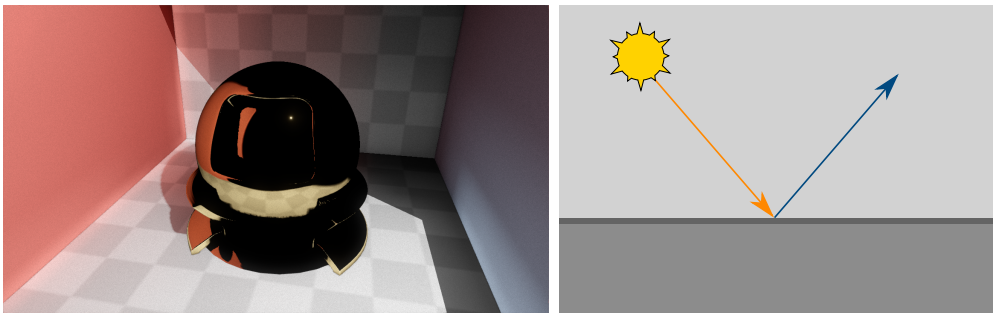
(a) Smooth diffuse



(b) Rough dielectric



(c) Rough conducting



(d) Smooth conducting

Figure 2.5: Four BSDF classes used in this thesis to simulate material behavior.

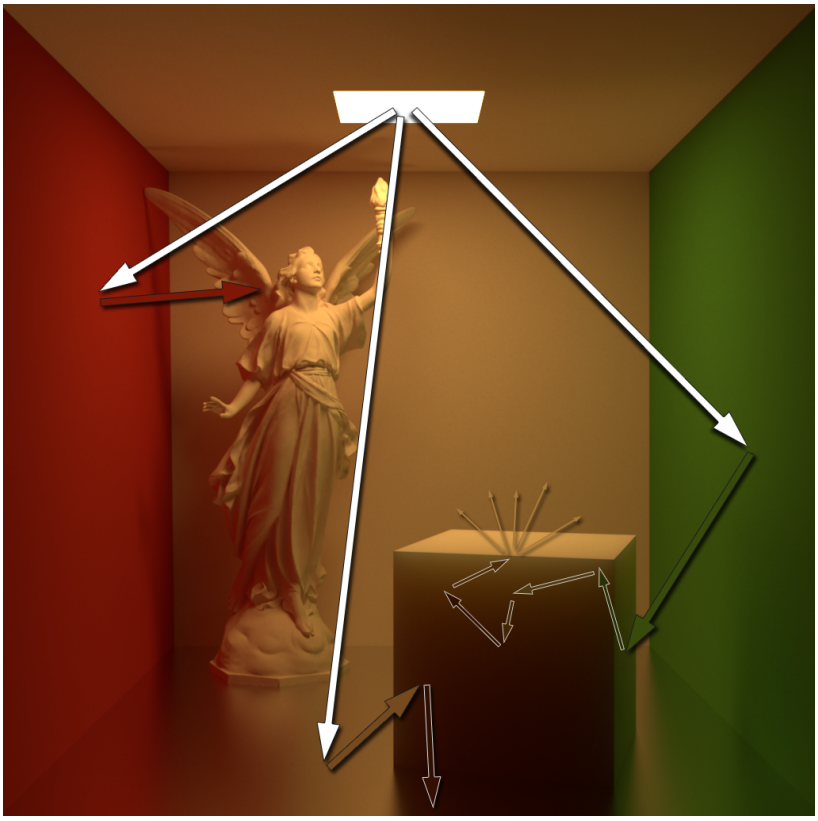


Figure 2.6: Global Illumination: Scattered light is reflected back into the scene, where it can scatter multiple times before all energy is absorbed by interaction with matter. Blocked indirect light can cause indirect shadowing (for instance visible above the Lucy figurine on the ceiling). Light may also scatter below surfaces (for instance in the small cube on the right) before exiting, causing a smoothed appearance.

surfaces within a vacuum, a path is a poly-line with vertices at each surface corresponding to a scattering bounce (a more complex case is presented in Figure 2.6). Edges connecting two vertices correspond to particles traveling in free space. A reformulation of Equation (2.7) within this new framework is called the *Path Integral Formulation*, initially developed by Spanier and Gelbard for Neutron transport [SG69] and introduced to computer graphics by Veach [Vea98]:

$$I_j = \int_{\overline{\Omega}} f_j(\overline{x}) d\mu(\overline{x}) \tag{2.24}$$

Equation (2.24) replaces the classical form of Equation (2.7) with a functional integral over an infinity of possible paths $\overline{\Omega}$[3] to compute the camera response I_j with respect to each pixel j. The integrand $f_j(\overline{x})$ is called *measurement contribution function*, encompassing the amount of light transported along a path with all given interactions in between.

$$f_j(\overline{x}) = L_e(x_0 \rightarrow x_1)T(\overline{x})W_e^j(x_{k-1} \rightarrow x_k) \tag{2.25}$$

f_j is the product of the emitted radiance $L_e(x_0 \rightarrow x_1)$ along the first segment of a path $\overline{x}$ of k vertices, the *transported* throughput of the entire path $T(\overline{x})$ and the sensor sensitivity or importance $W_e^j(x_{k-1} \rightarrow x_k)$.

For a more in-depth look into the path integral formulation I refer the interested reader to Chapter 8 of Eric Veach's dissertation [Vea98].

2.1.4.2 Monte Carlo Methods

In only very rare cases it is possible to create an analytical solution for $L(\mathbf{x}, \vec{\omega_o})$. In the majority of cases when synthesizing an image parts of the equation are not closed form functions and therefore need to be solved probabilistically. A useful tool is the Monte Carlo estimator. Consider the following integral:

$$I = \int_a^b f(x)dx \tag{2.26}$$

An *estimate* $\langle I \rangle$ of I can be obtained from a weighted sum of random samples as follows:

[3]Please note that to avoid confusion between hemispherical and path domain I use $\overline{\Omega}$ in contrast to [Vea98].

$$\langle I \rangle = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)} \tag{2.27}$$

Each sample $f(x_i)$ has relative probability weight given by a Probability Distribution Function (PDF) $p(x_i)$. The expected value of the estimator is:

$$E\left[\langle I \rangle\right] = \frac{1}{N} N \int \frac{f(x)}{p(x)} p(x) dx = I \tag{2.28}$$

For this reason $\langle I \rangle$ can be used to estimate the value of I. If the estimator is of exactly the value as the integral (i.e., $\langle I \rangle = I$) it is said to be *unbiased*. If however there is a non-zero difference, this value is called *bias*. Biased estimators may be a more reasonable choice than unbiased ones for example when faster convergence or less variance is preferred over reducing a systematic error. If the estimator is adaptive and bias vanishes in the sample size limit it is said to be *consistent*.

Monte Carlo estimators find immediate applicability in the Path Integral framework: Computing radiance is now an estimation problem over the domain of all possible paths which may have arbitrary length, but for reasons of practicality are usually limited to some upper bound. But as one can easily imagine, beyond light paths of very limited lengths, with each additional bounce the amount of computation needed to add up additional light grows to unmanageable proportions for real-time applications.

2.2 Real-time Rendering

To create the illusion of interactivity with a virtual scene, a renderer needs to synthesize images in rapid succession. Definitions for what constitutes the

right speed vary in literature, but a common understanding is that of *interactive behavior* (less than 1000ms) and *real-time* (less than 33ms).

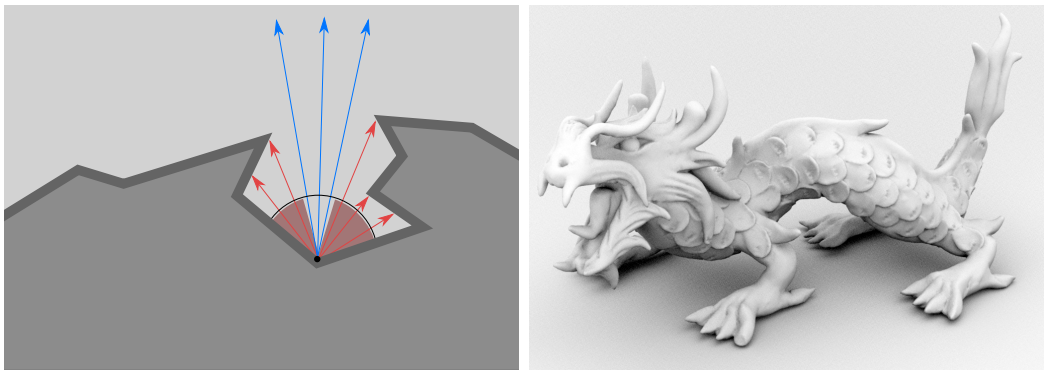
It is clear however that both budgets may require to cut down on the costs of correct light transport in order to produce an image on time. Options to do so include: Optimization of implementations, dedicated hardware support, better data structures, exploiting limitations of the human visual system, limiting complexity and above all algorithmic simplifications. The following section will review tools commonly used to synthesize realistic images in real-time.

2.2.1 Precomputed Methods

A first optimization when solving Equation (2.7) is to consider elements of the scene which do not change, such as static surfaces, objects with fixed spatial positions or materials or an unchanging incident light configuration. Because these elements stay constant for a certain time-frame (or the entire simulation), they can be computed in advance.

2.2.1.1 Lightmaps

Under fixed illumination conditions, static objects will always be lit the same and always have the same influence on their surrounding, i.e., shadows appear at the same positions and brightly lit surfaces will stay this way. For view-independent effects such as darkening due to shadows and diffuse scattering, these effects can be stored in *Lightmaps*, which are data structures (usually textures) created using for instance a raytracer and mapped/multiplied with the albedo value of static objects at runtime. Depending on the size of the scene, it may or may not be more efficient storage-wise to premultiply lighting with material response.



(a) Inside-out raytest. (b) A sample result.

Figure 2.7: Ambient Occlusion. (a) Self-occlusion is averaged and stored as an attenuation factor per surface point. (b) Ambient occlusion for a scene with the XYZRGB Dragon model.

2.2.1.2 Ambient Occlusion

Concave objects exhibit self-occlusion behavior, and if the surface does not change, the average self-occlusion A for each point on the surface can be calculated in advance. A visual representation can be seen in Figure 2.7(a).

$$A = \frac{1}{\pi} \int_{\Omega} V(\mathbf{x}, \vec{\omega_i}) \langle \vec{\omega_i}, \vec{\omega_o} \rangle_+ d\vec{\omega_i} \tag{2.29}$$

Here, $V(\mathbf{x}, \vec{\omega_i})$ is a binary value representing the visibility of $\mathbf{x}$ from or into direction $\vec{\omega_i}$[4]. The average value can be mapped onto the surface as an attenuation factor. When integrating light over the entire hemisphere Ω, Equation (2.7) is now simply augmented by A.

The resulting appearance is statistically correct under homogeneous white ambient lighting conditions, but can be used with many different scenarios as the result is visually appealing [Lan02]. Figure 2.7(b) features the XYZRGB-Dragon model computed with 64 visibility tests per surface point.

[4] AO methods are classified as inside-out or outside in tests depending on the direction of the test.

2.2.1.3 Precomputed Radiance Transfer

The computation of the integral in Equation (2.7) can be greatly accelerated by solving the convolution in another domain [SKS02].

As with many other integral transforms, a suitable new basis can be used to move both representations into a different space where certain calculations may be easier to perform. The dot product of the resulting coefficients of each function in a new basis Φ approximate the original integral. If the transformation into the new basis and the dot product can be performed in less time than the actual integration, this behavior can be exploited to accelerate its computation.

$$(f * g)(t) = \int_{-\infty}^{\infty} f(t)g(t-\tau)d\tau \tag{2.30}$$

Consider Equation (2.7): If we bundle both the Lambertian term $\langle \vec{n}, \vec{\omega_i} \rangle_+$ and the BSDF $f(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o})$ term into one function $T(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o})$ we can generate an integral transform of two functions. Furthermore, if T features no view-dependent effects, the function can be rewritten as $T(\mathbf{x}, \vec{\omega_i})$. For simplicity, we also remove the self-emittance term L_e in this example.

$$\begin{aligned} L_p(\mathbf{x}, \vec{\omega_o}) &= \int_\Omega f(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}) L_i(\mathbf{x}, -\vec{\omega_i}) \langle \vec{n}, \vec{\omega_i} \rangle_+ d\vec{\omega_i} && (2.31) \\ &= \int_\Omega L_i(\mathbf{x}, -\vec{\omega_i}) T(\mathbf{x}, \vec{\omega_i}) d\vec{\omega_i} && (2.32) \end{aligned}$$

In Equation (2.32), function T is called a *transfer function*. By separately convolving both L_i and T with a set of basis functions Φ, a set of coefficients can be computed representing the light transfer over a surface as a set of vectors or matrices.

$$t_c = \int_\Omega T(\vec{x})\Phi_c(\vec{x})d\vec{x} \tag{2.33}$$

$$l_c = \int_\Omega L_i(\vec{x})\Phi_c(\vec{x})d\vec{x} \tag{2.34}$$

Each signal can be reconstructed with its coefficients.

$$T(\vec{x}) = \sum_c^\infty t_c\Phi_c(\vec{x}) \tag{2.35}$$

$$L_i(\vec{x}) = \sum_c^\infty l_c\Phi_c(\vec{x}) \tag{2.36}$$

If Φ is an orthonormal basis, the dot product of n coefficients t_c and l_c reproduces an approximation $\tilde{L_p}(\mathbf{x}, \vec{\omega_o})$ of the original integral transform $L_p(\mathbf{x}, \vec{\omega_o})$ in Equation (2.32).

$$\begin{aligned}
\tilde{L_p}(\mathbf{x}, \vec{\omega_o}) &= \sum_c^n t_c l_c && (2.37)\\
&\approx \int_\Omega \sum_a^n t_a\Phi_a(\vec{x})dx \int_\Omega \sum_b^n l_b\Phi_b(\vec{x})dx && (2.38)\\
&= \int_\Omega L_i(\vec{x})T(\vec{x})dx && (2.39)\\
&= L_p(\mathbf{x}, \vec{\omega_o}) && (2.40)
\end{aligned}$$

If a set of surfaces for which the transfer function has been defined is rigid, material response such as subsurface scattering etc. do not change for incident light and self-occlusion remains constant. Under this assumption, the advantage of this method becomes clear: The transfer coefficients can be *precomputed*, thus making a band-limited approximation of the rendering equation available for real time rendering. Furthermore, the computation required to calculate $\tilde{L_p}$ is now decoupled from the complexity of T.

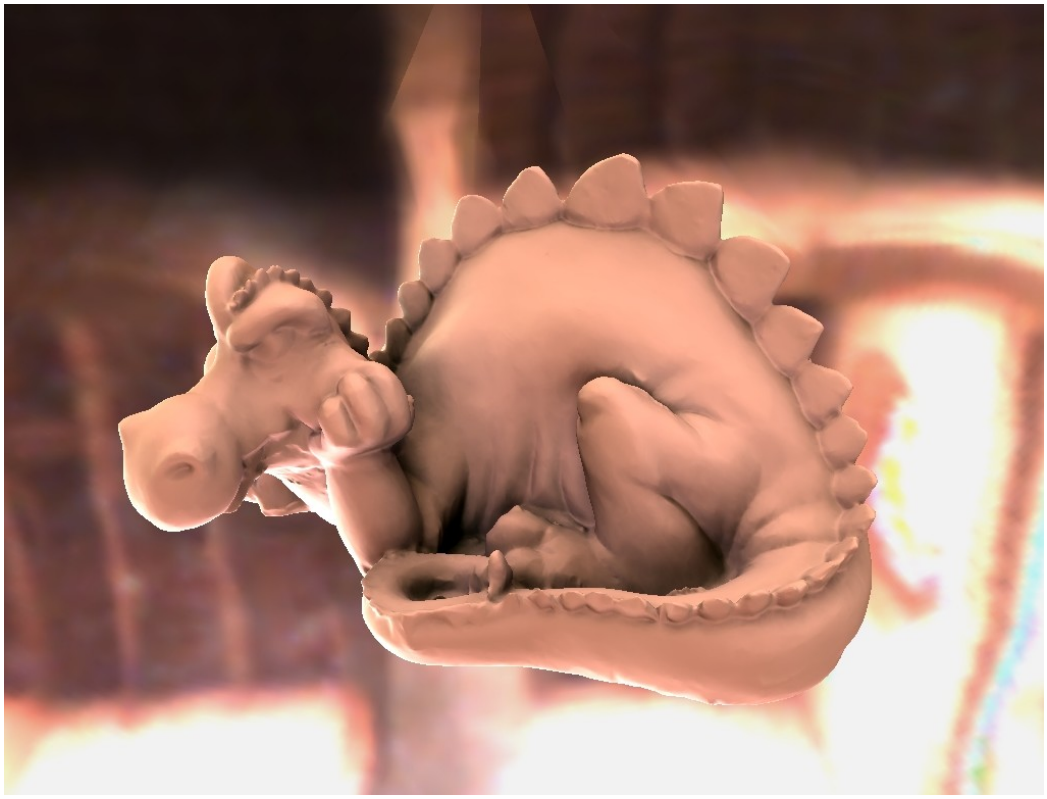
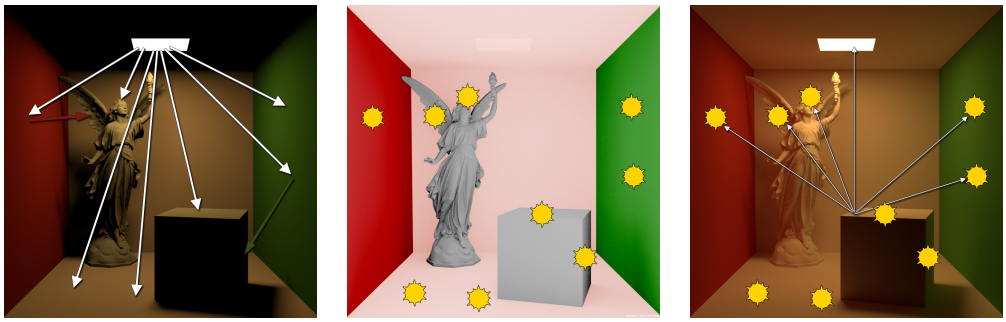


Figure 2.8: An image synthesized with Precomputed Radiance Transfer: The coefficients of the background image multiplied with the transfer yield smooth shading and self-shadowing.

A popular basis function for PRT is the real spherical harmonic function $y(\theta, \phi)$, which is conveniently defined on the domain of a unit sphere and can be easily implemented [Gre03]. One way to precompute the coefficients is to use a raytracer and sample the hemisphere of each vertex in an external tool or during initialization. A sample rendering of a ceramic dragon can be seen in Figure 2.8. Self-shadowing is preserved for small cervices on the surface.

2.2.2 Many-Lights Algorithms

A class of Global Illumination algorithms make use of caching techniques to accelerate bounce computation. These caching techniques usually operate under the assumption that a surface interacting with a light bounce is restricted to certain types of materials, most often diffuse. Popular exam-

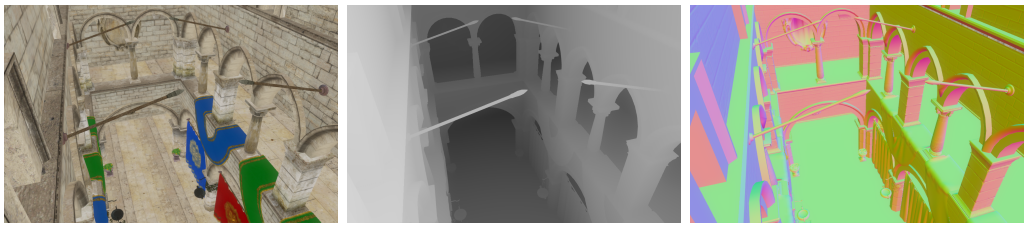


(a) Scatter (b) Create VPLs (c) Gather

Figure 2.9: Instant Radiosity algorithm overview. (a) Light scatters through the scene. No shading is yet performed. (b) At each vertex along a path, a VPL is generated. (c) Shading is essentially reduced an integration over all VPLs and direct light sources.

ples include Radiosity [GTGB84], a technique derived from earlier studies on heat-transfer exchanging diffuse energy between small patches of the scene, and Irradiance Caching [WRC07], a data structure saving sparse samples of computed irradiance. The general framework includes a *final gathering* step, in which illumination cached for various parts of the scene is gathered to accumulate the final reflected energy at each point in the scene. Many-Lights Algorithms use a deferred step to create light sources which represent indirect illumination.

Instant Radiosity [Kel97] is a caching technique operating with small point light sources that represent the current bounce. Instead of gathering light contribution along a path, at each vertex of the path a small *virtual point light* (VPL) is placed with the direction of the last surface normal the bounce interacted with it and the reflected wavelength. VPLs can be thought of as a form of caching. In a final gathering step, other surfaces integrate over the set of VPLs to accumulate indirect illumination. In the Path Integral framework, this can be seen as a bidirectional path tracing operation, where paths from a light source are constructed and vertices are replaced by VPLs, and afterwards surface points are connected as eye-paths of length 1. Because VPLs may be used to illuminate multiple points in a scene, this form of sub-path reuse is usually faster than generic bidirectional methods. An overview of Instant Radiosity can be seen in Figure 2.9.



(a) Albedo (b) Depth (c) Normals

Figure 2.10: A Reflective Shadow Map is a Geometry Buffer as seen from a directional light source.

$$
\begin{aligned}
L(\mathbf{x}, \vec{\omega_o}) &= L_e(\mathbf{x}, \vec{\omega_o}) + \int_{\mathbf{D}} f\left(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}\right) L_p(\mathbf{x}, -\vec{\omega_i}) d\vec{\omega_i} && (2.41)\\
&= L_e(\mathbf{x}, \vec{\omega_o}) + && (2.42)\\
&\quad \sum_{p}^{M} f\left(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}\right) G\left(\mathbf{x}, \mathbf{x}_p\right) V\left(\mathbf{x}, \mathbf{x}_p\right) E_p\left(\mathbf{x}, \mathbf{x_p}\right) && (2.43)
\end{aligned}
$$

The reformulation of Equation (2.7) into Equation (2.43) replaces the recursive integral with a sum over all light sources in the set $\mathbf{D}$ of emissive lights and VPLs p. Each VPL is weighted additionally by a binary visibility term V between the position of the surface point $\mathbf{x}$ which is shaded and the position of the VPL $\mathbf{x}_p$, as well as geometry factor G between both points which determines their geometric coupling.

$$
G(\mathbf{x}, \mathbf{y}) = \frac{\langle \vec{n_x}, \mathbf{y} - \mathbf{x} \rangle_+ \langle \vec{n_y}, \mathbf{x} - \mathbf{y} \rangle_+}{|\mathbf{x} - \mathbf{y}|^2} \tag{2.44}
$$

An elegant method to create VPLs in a rasterizer has been presented [DS05]: A Reflective Shadow Map (RSM) is a geometry buffer and an extension to regular shadow maps, adding normals and albedo. Assuming that pixels in a RSM represent diffuse reflectors one can use it to reconstruct VPLs with position, direction and flux. See Figure 2.10 for an example of an RSM.

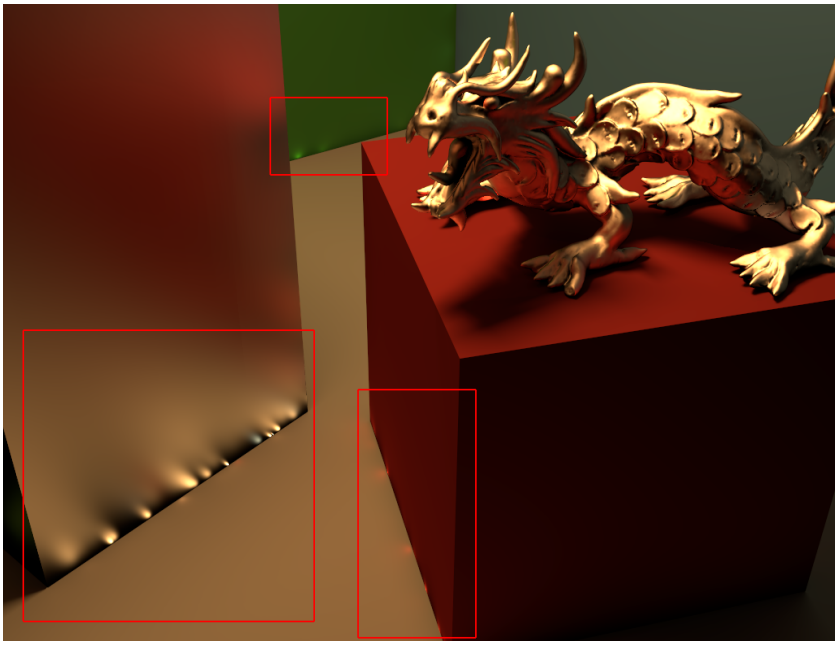


Figure 2.11: Instant Radiosity artifacts due to unclamped highlights from VPL singularities.

An essential open problem of Instant Radiosity is to find a good sampling strategy for VPLs, as derived methods typically add up to high evaluation costs in rasterizers due to significant overdraw or suffer from temporal incoherence when using a low number of indirect lights. Furthermore, although Equation (2.43) is fairly simple to evaluate, the geometry term G is unbounded and is the cause of local singularities from VPLs near a surface (visible as bright spots[5], see Figure 2.11). This artifact is usually countered by clamping the contribution of a VPL:

$$G_b(\mathbf{x}, \mathbf{y}) = \min\left(G\left(\mathbf{x}, \mathbf{y}\right), b\right) \tag{2.45}$$

The energy loss due to G_b needs to be compensated to avoid bias [NED11].

Global Illumination effects are often coherent in space. Several clustering methods exist to increase the amount of VPLs without hurting performance.

[5]These are sometimes referred to as *laser pointers*.

To combat overdraw, Dachsbacher and Stamminger [DS06] propose splatting VPLs with a clamped influence radius. Nichols and Wyman [NW09] furthermore exploit the low-frequency nature of indirect diffuse and very glossy reflections. They identify discontinuities in a scale-space of the final image to render splats at lower resolution where no sudden changes occur. Lensing and Broll further [LB13a] refine this scheme creating better splats with individual weights for geometry attributes and pre-selecting four VPL candidates for surface points. In a similar manner, Radiance Hints [Pap11] cache radiance in volumes using Spherical Harmonics. When computing indirect illumination, each surface point is shaded using a voting system to determine the four best radiance hints. Lightcuts find a horizontal *cut* through a hierarchy of VPLs and merge together sections of low variance, essentially replacing several VPLs with a brighter representative [WFA+05]. Prutkin et al. [PKD12] importance sample an RSM and combine several VPLs via k-means clustering into area lights. Tiled Shading [OA11] subdivides the image space into regular tiles. In a separate compute step, for each tile a list of influencing light sources is compiled, reducing the final shading cost. LightSkin [LB13b] in contrast concentrates appropriate indirect sampling on the surface of the shaded object. Sparse samples compute the influence of disc-shaped clustered Virtual Area Lights (VAL) and the contribution is interpolated according to similarity between geometric attributes.

A significant performance bottleneck comes with the evaluation of indirect visibility, which is often neglected in real-time methods. Other approaches include incremental computation of shadow maps per VPL [LSK+07] or the use of Imperfect Shadow Maps [RGK+08] (i.e., very sparse omnidirectional shadowmaps evaluated for each VPL and combined in a secondary step).

The main limitation of Instant Radiosity however is that it is applicable to diffuse and highly glossy surfaces only [KFB10]. The sparse sampling of paths necessary to maintain its efficiency advantage over regular Monte-Carlo evaluation is also a severe source of feature blurring. Higher number of VPLs can solve this issue, but at the expense of the real-time effort.

Further mechanisms can be found in two State-of-the-Art reports on Many-Lights Algorithms [DKH+14] and on current real-time global illumination methods [RDGK12].

2.2.3 Screen Space Methods

A popular domain to solve rendering problems is the screen space itself. Screen space methods[6] only consider geometrical information such as depth, normals and albedo on a per pixel basis. An early proponent to take advantage of this information is found in Luft et al. [LCD06], who use unsharp masking of the depth buffer of an image to enhance it with ambient shadows, emphasizing the spatial relation between objects. Similarly, Screen Space Ambient Occlusion (SSAO) uses the depth buffer to derive a simplistic ambient occlusion model per pixel [Mit07]. Since then, an abundance of optimized models have been proposed [BSD08, MSW10, HKM11, MOBH11, HBR+11, MML12, Tim13][7]. Screen Space Direct Occlusion (SSDO), a method introduced in [RGS09] that is similar to SSAO, samples the neighboring color buffer of a texel to gather near-field indirect illumination.

Many other effects can be computed in screen space, such as depth of field, bloom, atmospheric scattering, antialiasing, color warping, motion blur, ambient occlusion or localized reflections. Screen space methods derive their popularity from two major advantages: They decouple scene complexity from the computational cost of the effect, and they can be very easily integrated into existing pipelines. On the other hand, the same methods can only work with information readily available in screen space. Reconstructing geometry from depth and normals or raycasting in screen space have to rely on partially available information, as the backside of objects or geometry outside the boundaries of the screen space is not included.

Screen space methods therefore often augment an image for localized effects. Generally, as soon as necessary information cannot be derived from screen space buffers, lookups are deferred to other sources such as fixed local light-

[6] Sometimes in literature also referred to as image space methods.
[7] This represents only a small sample. Refer to [RDGK12] and [Aal13] for overviews.

probes or global environment maps (see for instance [TVB+14]). Another class of algorithms called *Deep Screen Space* reconstruct geometry information from several layers in screen space. These methods have the ability to detected backfacing surfaces, but still need special handling once they leave the screen space boundaries [MMNL14, NRS14].

An interesting class of algorithms consider augmenting images with second bounce reflections and are often referred to as *Screen Space Local Reflections*: Raycasting in screen space is used to determine reflections on surfaces which appear in the same image [TVB+14]. Like SSDO, this method can be used to compute localized global illumination. Because this methods is intrinsically not very different from regular raytracing, the same solutions for generating good sampling apply.

2.2.4 Visual Equivalence

Another category of simplifications can be derived from limitations in human perception of visual stimuli. A broad range of studies have been conducted on varying parameters and fidelity of materials, geometric complexity and lighting conditions and their effect on human perception. A particular line of work coined the term *visual equivalence* in perceptually based rendering, where images with visible differences appear identical to one another [RFWB07, RBF08]. In this thesis, I am most interested in the effects of approximations of material reflections and limited global illumination.

To cut down on the cost of computing GI shading we can limit the length of each path to a fixed number of bounces before connecting it to an emitter. In doing so systematic *bias* is introduced, which will manifest in certain types of artifacts discussed later.

The visual importance of indirect effects can be seen in Figure 2.12, where a scene is shown after n bounces of light before being directly connected to the camera. The most significant contribution in illumination after *direct* lighting (i.e., the second image where light bounces off one surface into the camera) is

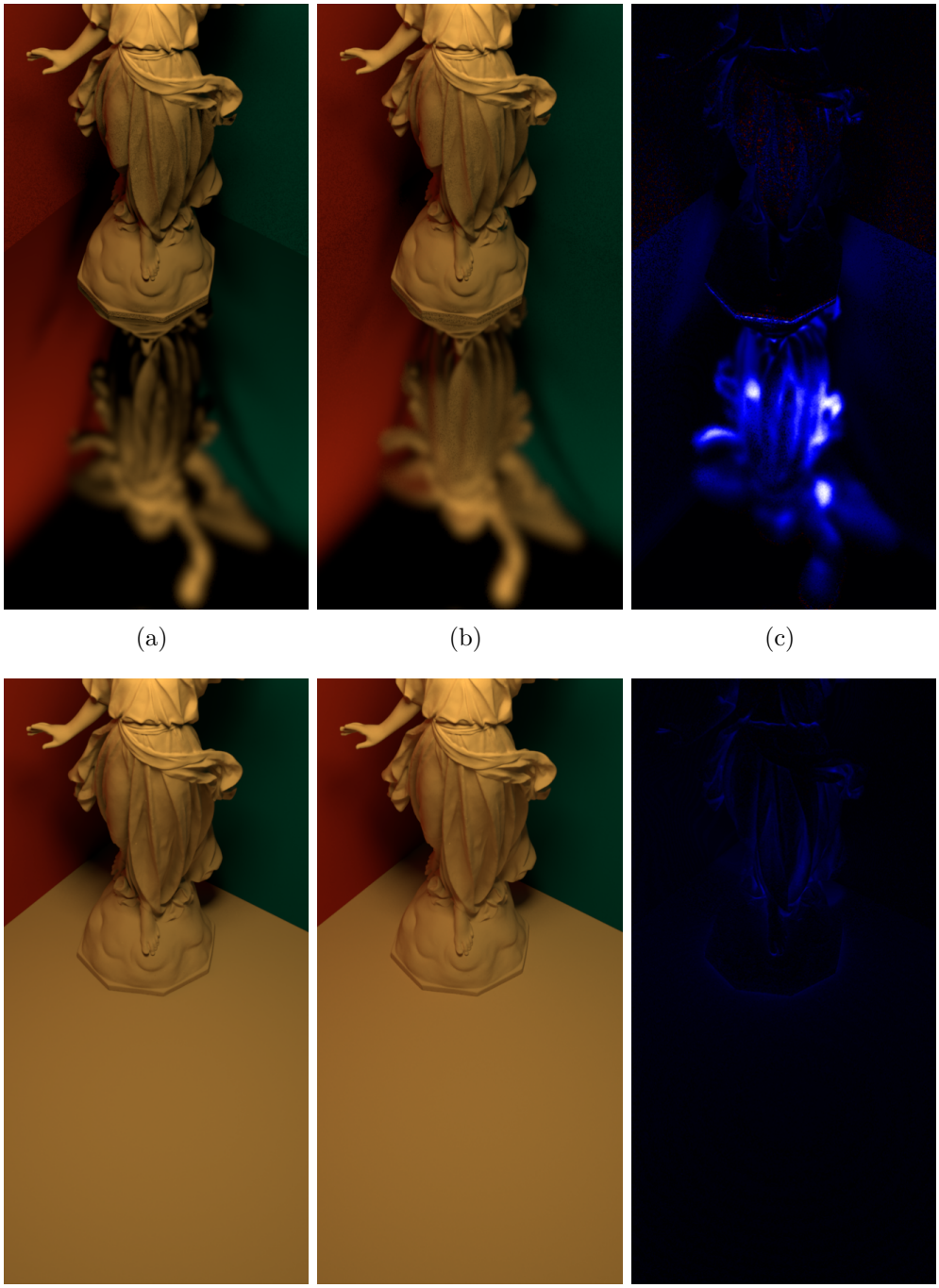
(a) 0 bounces (b) 1 bounces (c) 2 bounces

(d) 3 bounces (e) 4 bounces

Figure 2.12: Light bouncing around the scene: While the visible contribution of light indirection is decreasing with each bounce, renderers which stop short in the first three segments of the path (a-c) leave visible dark sports in the image.

visible in Figure 2.12(c), whilst successive steps do not add much additional energy to the scene. Motivated by this fact, prior publications chose to cut off paths at this length in favor of efficiency [TL04, DS05].

However, the circumstance that paths of length three are sufficient for most details in an image breaks down as soon as glossy or highly specular surfaces are introduced into a scene. In Figure 2.13, there is a drastic difference in the reflection: The reflected scene does not match the appearance of the figurine on top. Where indirect light illuminates otherwise dark areas, the glossy surface only reflects a directly lit image, thus reproducing the dark spots. In contrast, in a scene featuring only ideal diffuse reflectors this error



(a) (b) (c)

(d) (e) (f)

Figure 2.13: The mismatch (c,f) between a limited path length of three (a, d) versus a length of four (b, e) bounces is much more emphasized in the presence of glossy surfaces (a-c) than on ideal diffuse reflectors (d-e). Blue-White are positive, red-yellow negative errors. The difference images (c, f) show a three-bounce image subtracted from a four-bounce image.

is spread over a much larger surface and is therefore less noticeable. Because real scenes can often feature glossy or highly specular surfaces, paths in scenes which are not ideally diffuse everywhere should be extended to a length of four.

The impact of approximations on GI such as limited path length have been studied in recent publications. Křivánek et al. [KFB10] identify boundary conditions of parameters in VPL-based methods such as energy clamping and VPL counts, as well as materials which cannot be accurately reproduced using such methods. Guided by these boundaries, they propose a method to efficiently compute perceptually equivalent results. Their investigation indicate that the number of VPLs used in a scene is tied to geometric complexity of an object. Generally, geometrically complex surfaces tend to be more forgiving, as well as dielectric materials. Visibility and occlusion for glossy BRDFs are highly affected by the geometric complexity [KK07]. The study finds that objects with highly glossy BRDFs remain almost unaffected visually because of blurred scattered light. For scenes with smooth specular surfaces, larger area light sources equalize the amount of occluded and unoccluded samples on a surface and are therefore suitable for approximation (such as [GKD07b]). In a similar manner, Yu et al. [YCK+09] analyze the effect of visibility approximations on indirect illumination. The study compares Imperfect Shadowmaps [RGK+08], Directional Ambient Occlusion [RGS09], Ambient Occlusion and no visibility and finds that approximating visibility for indirect illumination yields renderings which are very similar to reference images. Beyond this insight however, even approximations which produce larger discrepancies to the reference are often perceived to be realistic.

A limited analysis of global illumination algorithms in the context of Augmented Reality from Hattenberger et al. [HFJS09] surveys Whitted Raytracing, Photon Mapping and Path Tracing. The study concludes that algorithms which include global illumination effects outperform other algorithms in terms of perceived realism, however observers tend to rank images with high-frequency noise lower than others. This is an interesting and important observation which can lead to somewhat paradoxical results: In the study, images generated by path-tracing which included indirect illumination but

contained high-frequency noise artifacts were ranked lower than images with direct illumination only.

2.3 Augmented and Mixed Reality

Augmented Reality is a mixture of real and virtual elements, situated in the section of the continuum in Figure 1.2 where reality is enhanced rather than virtuality. In this section, I will review the basics of an Augmented Reality system: Display technology, current tracking mechanisms for geometric registration and reconstruction algorithms which extract and gather information about the geometry and layout of reality. While all of these elements do not contribute to an underlying rendering implementation, they highly influence the way how algorithms shading an object or the surrounding can or must be adapted.

2.3.1 Camera & Display

There are different ways to augment reality with new information. A subgroup of this technology is called *Projection Mapping* or *Spatially Augmented Reality* (SAR), where real in-place objects are relit to change their appearance. Here the place-holder defines the geometry which is to be augmented with a new appearance, and is usually a white, ideally diffuse object which only minimally influences the projected appearance. A crucial challenge with SAR is the correct rectification and mapping of projected appearances. Because one view of the scene for a given projector might not map the visible area (for instance, one view can only map the front side of an object), a second challenge is the correct registration of multiple projection devices onto a single object with minimal, seamless overlap, especially for high-frequency content such as highly specular materials [BTS13]. Illuminating real surfaces can lead to unintended scattering between them. A *reverse* GI solution has to be computed to compensate for these artifacts [SYC10, SCCN11].

The second class, where entirely virtual objects are placed into a scene, uses a display as the *window-to-the-world*. A simple setup consists of a camera, which records the real scene, and outputs the image as a background to the display, such as a modern tablet computer. More advanced setups include head-mounted displays (HMD) such as *AR-glasses*, which can be further distinguished into two categories: Optical see-through displays with transparent display technology such as the *Microsoft HoloLens*, or stereo displays with mounted cameras such as the AR-Rift [SJS14].

Optical see-through displays offer the advantage of being unobtrusive compared to HMDs. A major downside however is the lack of true occlusion: An augmenting pixel cannot completely occlude the real background, i.e., it cannot properly turn to true black. Augmenting objects appear as ghosting, half-transparent surfaces, which can severely affect the perceived level of realism. Furthermore, current optical see through displays struggle with limited field of view. Augmenting arbitrary surfaces in the actual viewspace of the observer is often not possible if the right viewing angle is not met.

On the other hand, HMD stereo setups such as the AR-Rift [SJS14] or regular display based technologies such as tablet computers do not suffer from these problems. They do however introduce aliasing either on the (near-eye) display, which can furthermore degrade image quality through various artifacts of its own, or from artifacts of the cameras mounted on the front. These include color shifts and distortions, motion blur, defocus blur, grain and high ISO noise, chromatic aberration[8] and lens distortion. To address these artifacts, one can either render an equally distorted image with matching artifacts, or try to revert these issues in the original image before augmenting it [KM08, KM10, KSF10, KTPW11, OHSG12, PLW12].

[8]This is due to the dispersion of light on the lens due to different wavelengths discussed earlier.

2.3.2 Geometric Registration

To reliably position a virtual object into a real environment the virtual camera or window-to-the-world needs to be *geometrically registered*. Geometric registration is the process of transforming multiple data sets into a common coordinate system, which can be done manually or taken up on by an automated method. Geometric registration is subdivided into two phases: Initialization (i.e., deriving the initial position in real space of a virtual camera) and tracking (i.e., the process of constantly updating the cameras position relative to the last frame).

Algorithms may be classified into several categories, but a common one for Augmented Reality is to divide them into intensity-based and feature-based algorithms, which search a *target image* for a *reference image*. Intensity-based algorithms compare extracted patterns in the target image with the reference image of the pattern using correlation metrics and – if registered – treat the center of the corresponding pattern as a feature point. Feature-based methods detect correspondence between a number of *features* (i.e., local patches which are distinctive enough to be recognized) in the reference and the target. Based on this point-to-point mapping, a transformation can be established.

Depending on the device used for tracking, additional built-in sensors may be used to either help identify the initial location of the camera or support and stabilize the tracking method [BS09]. These include Global Positioning System (GPS), accelerometers and gyroscopes. A valid assumption about the position of objects for instance is their orientation with respect to gravity [KB12].

Initialization is in its most basic form achieved by searching the image for a recognizable marker, which can be a simple black-and-white pattern or an image. A naive approach to geometric registration called *marker-tracking* is to re-initialize every frame, which avoids further implementing a tracking scheme. An open source implementation can be found in the publicly available ARToolkit [KB99].

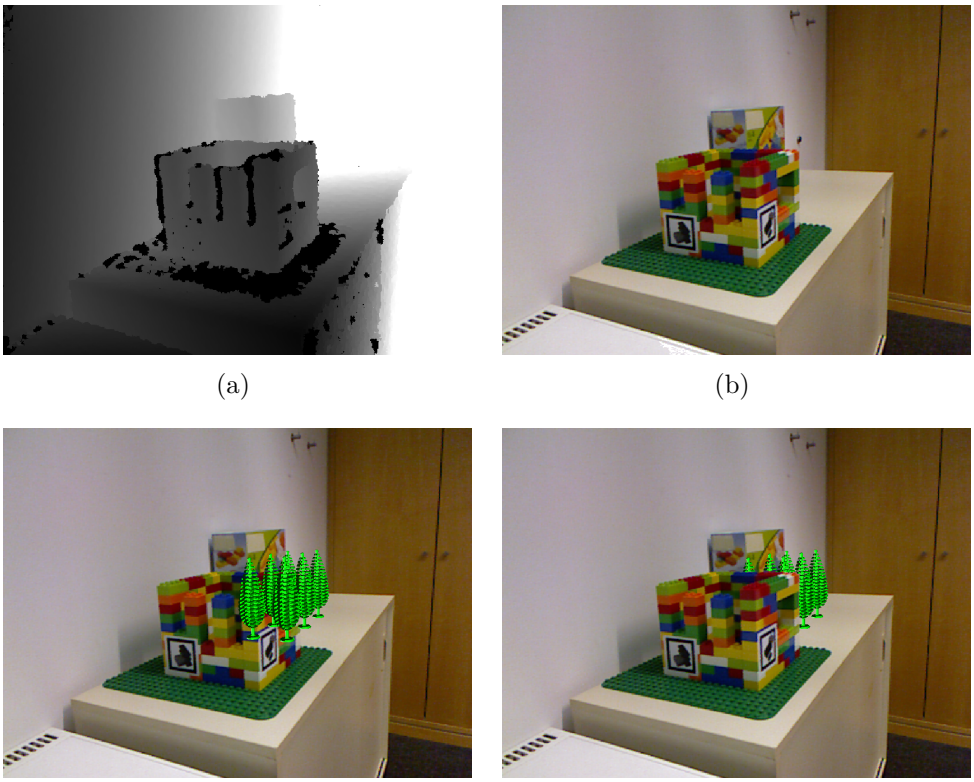
The image is converted to a gray-scale color scheme and then binarized. Afterward, contours are determined for blocks in the image, where any contour of four vertices may hint at a marker. To be certain, one has to check whether or not the contour is convex. The block is then reprojected to match the tracked image extents of the sample marker which was provided to initialize the image.

Tracking usually refers to methods which, given an initialized position, determine the linear transformation of a tracked object in space in consecutive video frames. By either analyzing the optical flow [LK81], finding similar contours [WVS05, Wue08], filtering particles [IB98] or repeatedly registering an image feature (such as marker tracking), the object of interest can be *tracked* continuously. Methods which rely on adaptively collecting new image features such as optical flow methods have the advantage that they can *loose* the object from the current view and do not need to reinitialize the image once the view returns to the correct orientation and position.

2.3.3 Reconstruction

Positioning a virtual object into a real context is a necessary, but not fully sufficient condition to create a believable mixture of both worlds. The real world position only allows the virtual object to be inserted on top of the real background image without any regards to lighting conditions or geometric occlusion (see Figure 2.14(b)). A renderer fusing the image needs to know where surfaces are in the real world and where lights are positioned in order to cast shadows from the virtual object onto real geometry, occlude the virtual object with a real world one etc.

Depending on certain conditions under which the AR application will operate, manual reconstruction of this information is a feasible option. For instance, if the scene is entirely static and the are of interaction limited to real world boundaries, a pre-reconstructed model of the real world can be used. It is entirely possible to even limit this reconstruction to the near boundaries of a virtual object instead: Shadows can be projected onto a patch beneath the object when the shadow projection range is limited. In this case, the only



(a) (b)

(c) (d)

Figure 2.14: Scene reconstruction from depth. (a) Raw depth image. At black pixels the Kinect depth camera could not acquire depth values. (b) Raw color image. The square markers on the brick model are used to track the camera pose. (c) Color image augmented with virtual trees (without occlusion). (d) Correct occlusion handling by mapping the depth measurements on the color image.

assumption one has to make is that a real surface beneath the virtual object is always planar when repositioning it.

When the augmented surrounding is frequently changing however, manual reconstruction is not an option. Geometry and light conditions need to be determined by an automated solution. All reconstruction processes can be supported by additional sensing hardware.

2.3.3.1 Geometry

Reconstruction of real surface geometry is a crucial component for synthesizing images with new objects. Reconstructed surfaces provide knowledge of light propagation in a scene, which in turn is necessary to give an object a spatial context. Shadows for instance can be projected once a surface is known, or occlusions between real and virtual objects depending on their spatial relation.

A straightforward method to reconstruct surfaces of a real scene is to do it by hand. A pre-reconstructed model of the scene can be geometrically registered alongside an augmenting object with fiducial markers or by any other means. Karsch et al. [KHFH11] present an image synthesis system in which the user interactively roughly identifies planar areas. Using edge detection and user annotations, planar surfaces are reconstructed. The authors further automatize the pipeline by inferring depth cues from the image to reconstruct planar surfaces without requiring user interaction [KLK14].

Many recent publications build on the popular Microsoft Kinect RGB-D camera or similar devices reconstructing depth from either structured light or stereo views. As can be seen in Figure 2.14, rudimentary scene geometry reconstruction which rely only on screen space information can derive all necessary information from one depth buffer (Figure 2.14(a)). The background image in Figure 2.14(b) and its regular augmentation in Figure 2.14(c) can now properly react to occlusion of real geometry in Figure 2.14(d). Because single-image depth reconstruction only provides an incomplete geometric view of the scene, iterative methods have been presented to update static scenes from multiple views.

KinectFusion [IKH+11] uses SLAM to geometrically register a Kinect camera. A static scene can be *scanned* with the depth sensor while new depth data is fed into a global voxel structure, systematically updating a complete mesh of the scene. Because of the low quality depth reconstruction from the Kinect, noisy artifacts reduce the quality of the captured mesh. Chatterjee et al. [CJG12] investigate filters to create smooth surfaces undisturbed by sensor noise.

A similar recovery approach from photographs by Fuhrmann and Goesele [FG14] registers a dense set of photographs by matching features found between photographs of an object taken from different angles. Common features are extracted into a common depth map, which is filtered to fill holes in the reconstruction.

2.3.3.2 Light Sources

To support coherent illumination for augmenting virtual objects in relation to their real surrounding, it is crucial to determine where real light comes from. Natural illumination in real environments is often complicated, making proper analysis difficult for both humans and machines. However, one may infer illumination conditions by carefully observing several cues in a single image:

- Objects cast long shadows when a light source's angle of incidence is low.
- Shadow borders blur out if a light source either has a large area or is close to an object.
- Diffuse surfaces behave like Lambert reflectors, with the brightest spot oriented toward the light source.
- Local light sources can be identified from the varying reflected brightness on complete surfaces, which decreases with a factor $\frac{1}{m^2}$.
- Caustics spread out on the mediums opposing end when viewed from the light source's position.
- Specular highlights can be used to identify the exact position of a light source.

Various publications in the scientific literature have used one or more of these cues to identify real light sources in single images. However, this task without known surface properties becomes a dual problem of estimating reflectance

properties of real surfaces and then deriving knowledge about the scene lighting.

In [BRD14] the authors segmentize the image into objects with distinct material properties and find the most reflective one. Using known spatial relations of nearby objects in the image, the estimated material properties of the selected object and the observed radiance off the surface of the selected object, they recover a reflection map which has higher detail in the recovered reflection of known objects nearby, and – depending on the surface roughness – corresponding detail in the distant estimated light.

Arief et al. [AMH12] detect a single point light sources by correlating shadow shapes with a perspective mapping of their tracked objects.

An extensive study to reconstruct natural illumination in outdoor scenes has been conducted by Madsen et al. [Mad03, NM07b, NM07a, MN08, Mad08, ML10, ML13]. The authors propose to segmentize the image and extract shadow shapes and, based on several indicators (for instance GPS position and daytime) correlate the shadow shape with the current position of the sun to gather illumination conditions in the segmented area.

Sorbier and Saito [dSS14] incrementally reconstruct the environment from a Kinect and render an environment map for a virtual object augmenting the scene, which can be used to reproduce reflections on its surface.

Knorr et al. propose to use illumination captured from human faces [KK14]. Mobile devices often feature back-facing cameras which can be used to track and sparsely sample reflected illumination. A set of precomputed radiance transfer functions of sample faces under varying illumination is used as a basis. Through a least-square approximation of the samples and the approximated radiance transfer functions the original illumination can be reconstructed.

Gruber et al. [GRTS12, GLS+14] adaptively raycast on a signed distance field reconstructed with KinectFusion to create radiance transfer functions for surface sample points. With the help of these functions, the observed radiant exitance B can be solved for the incident light. Surfaces are assumed to be

diffuse reflectors, and the incident light is assumed to be white to ignore potential ambiguities of incident and reflected wavelength.

In a similar manner Xing et al. [XZL+13] and Liu et al. [LQX+09] reason about the illumination conditions using a basis of skylight and sunlight. A linear equation system of combined unknown sky and sunlight conditions is set up to solve for the incident light, given known surfaces and their diffuse coefficients.

Lopez-Moreno et al. [LMHRG10] extract silhouettes from segmentized objects. Assuming the normal along the silhouette lies in the image plane and that the object is convex, they extract azimuthal angles for light sources by observing the change of intensity along the silhouette and then find zenith angles by sweeping from silhouettes to the interior of the object. Leveraging the tendency that the human visual system perceives objects as correctly lit as long as the illumination is locally consistent, variations in lighting for different objects do not affect the final perception.

In an effort for more robustness, Lalonde et al. [LEN12] show how to combine three cues in the image – shadows, shaded surfaces and segmented sky – to estimate illumination conditions. Additionally, they combine the result with a data-driven prior to compute the final result.

Karsch et al. [KHFH11] interactively request the user to identify lighting cues (i.e.,different zones in the image), such as unoccluded windows or area lights.

These cues are not necessarily reliable. Many natural scenes can have very different sets of cues (if any at all) available for analysis. Further complications may arise from misleading shadows (i.e., multiple blurry shadows or simply washed out blobs from ambient lighting conditions) or highly reflective objects producing conflicting results in the observed lighting conditions, which break the assumptions made in estimation algorithms (such as diffuse reflectivity on all surfaces) needed to extract reliable segments in the image. Other image-based approaches therefore use real-world objects under controlled conditions to reconstruct real illumination conditions.

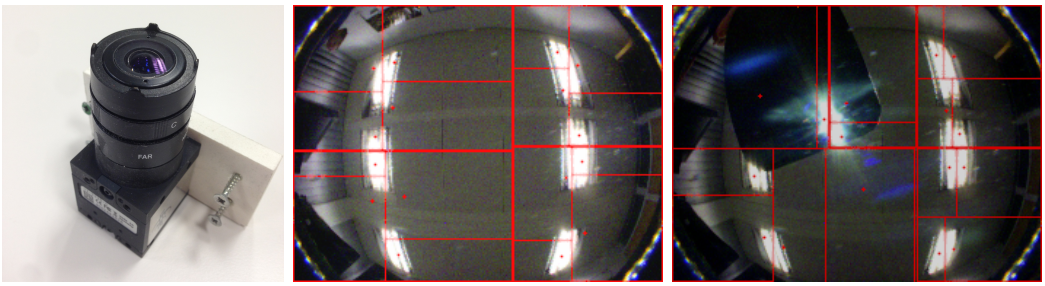
The most popular method to capture real or *natural lighting conditions* is directly related to reflection mapping [BN76a]: By capturing a highly specular sphere, often referred to as a *light-probe*, to reconstruct illumination conditions around an object. Usually, the light configuration is assumed to be distant to map it onto more complex surfaces with different spatial positions [Wil83, MH84]. Debevec uses light-probes to recover [Deb98] a high-dynamic range reflection map[9], which is then mapped onto a simple box model of the scene. Sato et al. recover [SSI99, SSI03] illumination conditions by analyzing variances in irradiance off shadowed regions in an image.

Many different methods exist to reconstruct point-light sources from light-probes or fish-eye lens cameras. Nowrouzezahrai et al. [NGM+11] factorize a low-order spherical harmonic representation of a light-probe into a directional and a global component. The directional component can be used as regular point light while the global term is applied using matrix radiance transfer.

A range of methods subdivide environment maps of distant light into regions of equal energy to find better samples. In [ARBJ03] Agarawal et al. use a Hochbaum-Shmoys algorithm to stratify an environment map. In [CETC06] the authors partition an environment map based on a BRDF and use a Summed Area Table to warp a uniform distribution into an important one. Viriyothai and Debevec [VD09] sample an environment map using variance minimization techniques on a median cut algorithm, while Ostromoukhov et al. [ODJ04] use a Penrose tiling scheme. Kollig and Keller use a Voronoi tessellation associated with directional properties of light sources [KK03]. Clarberg et al. propose to build a tree for uniform samples according to the probabilities supplied from a Wavelet analysis [CJAMJ05].

Karsch et al. present an automatic system reasoning about surfaces and partially unknown illumination from single images using a trained data set [KSH+14]. By assuming that two photographs with similar appearance have a similar illumination environment, they can pick an estimate from a database rather than reconstruct the actual environment, while Gibson et al. [GHH01]

[9] Referred to as *radiance map* in the paper.



(a) UEye UI-2230-C (b) Scene subdivision (c) New subdivision after changed conditions

Figure 2.15: Extracting 16 representative light sources from the image of a UEye UI-2230-C camera with a fish-eye lens: Lights are placed at the centroid of each region, which is determined by an intensity gradient.

solve a dual inverse rendering problem with representative *virtual light sources* in unknown regions, iteratively converging on the estimated emittance.

Finally, single important light sources can also be tracked with markers or registered using special sensors.

2.3.3.3 Materials

Deriving parameters for BSDF models or acquiring dense measurements for data-driven models has been the subject of extensive research [YDMH99, DvGNK99, MWLT00, MPBM03, GHH01, LKG+03, KSS+04], and more recently [MMS+05, PHD06, KKT11, WZT+08, SWRK11, TFG+13, CDP+14, KF14]. The scientific literature also contains reconstruction specific to human skin [DHT+00], food [SWR+11], cloth [SSK03] or metallic paint [RSK09].

Capturing reflectance from a sample is often done with a dome setup of several nodes of pairs of cameras and light sources [MPBM03, SWRK11, SSW+14]. The material is captured multiple times (usually in the thousands) under varying illumination from multiple viewing angles. The captured reflectance can either be a high dynamic range image of the sample to acquire a Bidirectional Texture Function (BTF) or, when capturing homogeneous materials, the scattering behavior of the reflection of a laser beam.

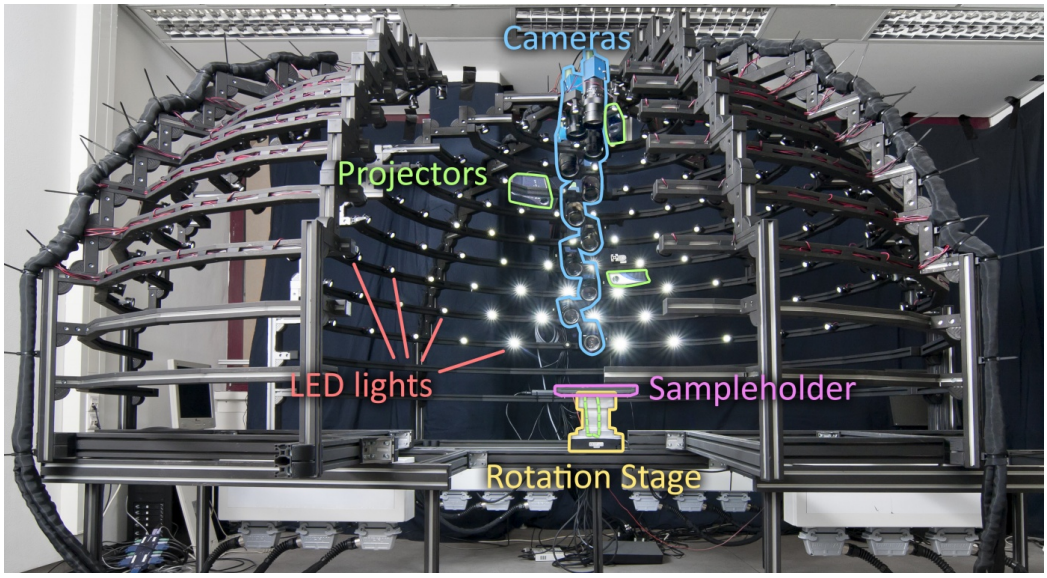


Figure 2.16: A dome setup to acquire a material sample. Image courtesy of Bonn University [SSWK13].

Data-driven models do not reflect inaccuracies of a parametric description of a certain material, but generally amount to very high data volumes after measurement. Due to the high coherence within the most data sets[10] compression schemes lend themselves well.

Several basis functions have been proposed to compress BTF: Haar Wavelets [CBP07], Principal Component Analysis [MMK03], Decorrelated Full Matrix Factorization [Mül09], Spherical Harmonics [WAT92] etc. For further details on BTF compression, I refer the interested reader to a state of the art survey [FH09].

An example of such measured data is the MERL BRDF database [Lab06, MPBM03] and the BTF database of Bonn University [Bon14]. A tool to compare measured data with simulated parametric models is available from Disney [Wal12].

In AR environments however complicated setups and analysis of surface properties are not feasible. Reasoning about surface properties has to be done in very few frames and even only partially determined lighting conditions.

[10] Consider for instance diffuse reflectance off a homogeneous surface under varying incident angles.

Problems of this nature are attacked with *inverse rendering* [Mar98]. Yu et al. [YDMH99] determine reflectance properties of a parametrized model from multiple high dynamic range, geometrically registered photographs of a scene using nonlinear optimization. In contrast Gibson et al. [GHH01] create a dual solution for partially unknown illumination and unknown reflectance properties. By placing *virtual light sources* into unknown regions, they split up incident light L into two sub-solutions and iteratively refine estimates for reflectance and virtual light sources until a final lower difference threshold has been reached. Using a complete reconstruction instead, Knecht et al. [KTTW12] can estimate diffuse surface reflectance with the help of captured light sources in real-time.

Karsch et al. [KF14] estimate spatially varying parametric materials of five degrees of freedom from single images with unknown shape and illumination using a prior set of HDR images for illumination. Reflectance is decomposed into two materials and weighting coefficients.

2.4 Further Reading

Because real-time global illumination in Augmented Reality systems is an interdisciplinary field, a large body of scientific literature quickly accumulates for the different sub-tasks. The following paragraphs give a short list of materials which contain surveys and overviews of the different topics.

Real-Time Global Illumination A great introductory reading into GI and different formulations is Eric Veach's dissertation on the Metropolis Light Transport algorithm [Vea98], which provides a solid theoretical understanding, as well as the book *Advanced Global Illumination* [DBBS06] by Dutre et al. Furthermore, the work of Pharr and Humphreys [PH04] and Hill et al. [HMD+14] provides a good basis for physically-based rendering. In-depth insights into current research on light transport can be gathered from the SIGGRAPH course on *Advances in Light Transport Simulation* [KKG+14]. An overview of Many-Lights Algorithms that provide a shortcut through

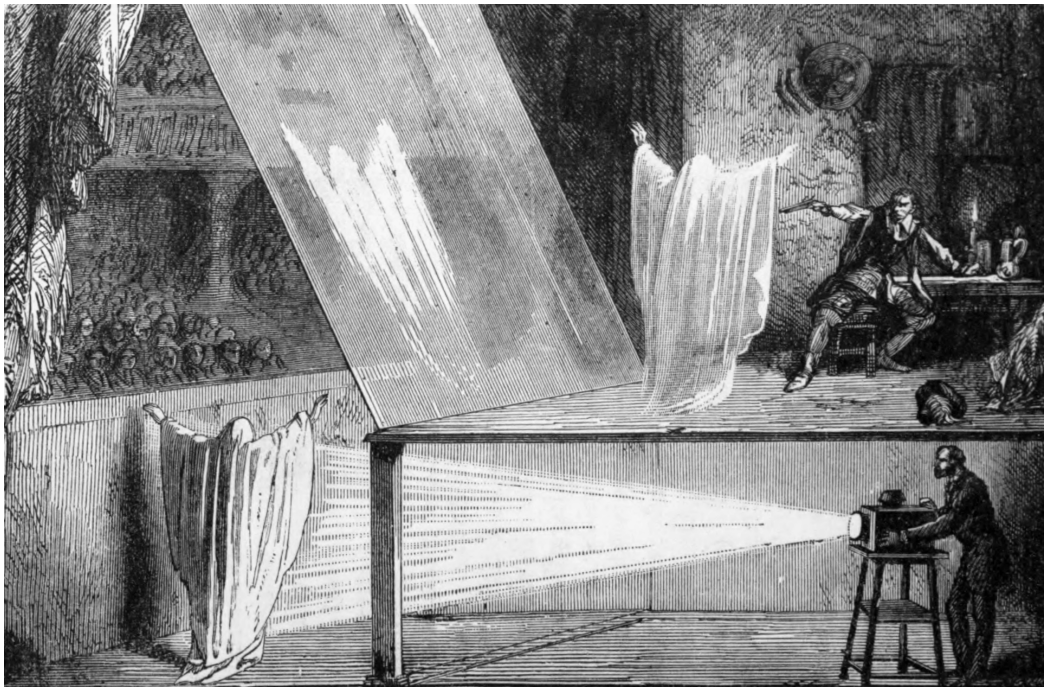
intensive GI computations is given in [DKH+14], whereas some of the condensed applicable results to real-time rendering are summed up in [RDGK12], including discussions of other GI frameworks such as precomputed methods, Discrete Ordinate Methods or Finite Elements. Further material on practical rendering techniques are found in the SIGGRAPH course notes of *Advances in Real-time Rendering* [TKD+14].

Relighting The process of *relighting* a scene changes an already set appearance, for instance of a real photo, to another and is a necessary tool to insert light bounces and shadows of virtual objects into real scenes in Augmented Reality. *RESHADE* was a project at the University of Vienna focusing on Mixed Reality relighting techniques. Publications spanning from 2010 to 2013 contain valuable knowledge about the entire relighting pipeline [Tra09]. Similarly, *RayEngine* is a project at the University of Vienna focusing on *High-Quality Real-Time Global Illumination in Augmented Reality*. Publications include important contributions in the field of relighting using Path Tracing [KK10]. The SIGGRAPH 2008 course material of *High dynamic range imaging & image-based lighting* presents a complete pipeline from capturing to image-based relighting of virtual avatars [WRD08].

Reconstruction A survey of publications up until 2003 in the inverse rendering domain can be found in [PP03]. The authors discuss several publications which deal with either inverse illumination problems (i.e., the recovery of incident illumination), inverse reflectometry problems (i.e., the recovery of reflectance properties of surfaces) or the combination of both. A listing compares algorithms with their constraints and initial assumptions about the scene. A State-of-the-Art survey on compression of Bidirectional Texture Functions can be found in [FH09]. Next to a BTF database [Bon14], Bonn University has extensively researched their recovery.

Chapter 3

The Delta Radiance Field



3.1 Introduction

In his 1864 letter to Richard Phillips titled “Thoughts on Ray Vibrations”, Michael Faraday outlined the profound idea that light propagates through

space with vibrating waves along *lines of force*. In this regard light should be interpreted as a *field*. This idea was later formalized by James Clerk Maxwell in a set of partial differential equations. A classic paper by Arun Gershun [GMT39] on radiometric properties in three dimensional space coined the term *light field*.

In geometric optics a simplified approximation to the electromagnetic field for light traveling through space and scattering is the five dimensional plenoptic function L, also called the radiance field:

$$L : \mathbb{R}^3 \times S^2 \rightarrow \mathbb{R}^3 \tag{3.1}$$

For a point in space $\mathbb{R}^3$ and a direction on a sphere S^2 a mapping exists to radiant flux (often in linear color space RGB $\mathbb{R}^3$). In contrast to a function defined on surfaces [Kaj86], like any other field L defines radiance flowing through each point in space from every direction.

Global illumination algorithms usually limit themselves to computing radiance at surface points, hence called *surface radiance*, whereas radiance computed in a field at arbitrary positions from any direction is called *field radiance*. This definition lends itself to find a mathematical description of *change* on existing radiance.

In this chapter, I will formulate a novel mathematical framework within which the next two chapters will operate. An analysis of the current standard practice to extract and relight a real scene – Differential Rendering – is followed by my definition of a *delta transfer operator*. This operator removes the severe computational burden when extracting changes in illumination between two scenes. It provides the necessary basis to efficiently adapt GI algorithms which can simulate effects which haven't yet been demonstrated in real-time AR, which will be presented in the following chapters.

3.1.1 Related Work

A pioneering work from Fournier et al. [FGR92] uses Radiosity to compute a global illumination solution for a virtual object in a reconstructed scene. The accumulated Radiosity result is used for virtual objects while for real objects only the additional delta is added. The formulation of the Delta Radiance Field builds the mathematical foundation with which several other methods closely relate to this approach.

Dachsbacher et al. [DSDD07] avoid explicit formulation of visibility in the transport operator by introducing a compensation factor called *antiradiance* or *negative light*; a new *unoccluded transport* operator is introduced. They test this new scheme using two iterative solutions working on patches. The relighting approach formalized in the next section implicitly contains antiradiance.

Similarly, Loos et al. [LNJS12] introduce Delta Radiance Transfer, extending their previous work, Modular Radiance Transfer, to include fine-scale transport from small objects which is ignored by the low-dimensional transport operators of their previous method. To recover the lost energy difference, they introduce two operators: *delta reflection* and *delta occlusion*, both of which either add positive or negative radiance to the scene to correct the final image. Delta Radiance Fields relate closely to this idea to correct for missing energy in a scene which hasn't been computed yet instead of being suppressed by low-frequency operators.

A system presented by Cossairt et al. [CNR08] records a scene which is divided by a lens array into a real part and a virtual part. A secondary camera records the real scene through the lens array to reconstruct a light field of the real scene, which is used to transfer real light onto virtual surfaces. Likewise, the virtual scene renders a second light field which projects virtual light onto real surfaces. By iterating between these two reconstructions back and forth, multi-bounce global illumination is achieved. Because of the lens array interface, fidelity of the reconstruction and freedom of movement of objects is limited, and virtual objects cannot shadow real surfaces. The framework presented below is inspired by these two interacting fields.

Grosch observes in [Gro05, Gro07] that a naive differential relighting is too expensive when rendering with more sophisticated shading algorithms such as photon mapping. Instead of employing the usual differential approach with two complete renderings, the main idea is to shift the differential into the photon map, thus avoiding the major drawback of rendering two largely identical images to extract the difference in illumination. The next section presents a theoretical generalization of this idea for global light transport.

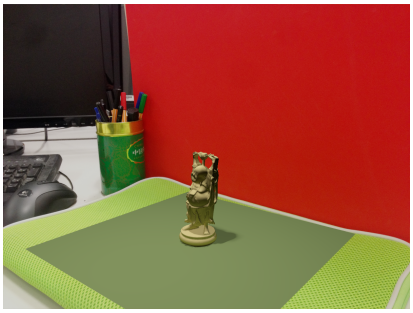
3.1.1.1 Differential Rendering

Early prototypes in Augmented Reality did not relight real scenes when introducing new objects. Instead, real surfaces were reconstructed as close as possible to the original and shaded like all other virtual objects. The result was then simply overlaid onto the original surface in the augmented image. Because of micro-scale structures, material and color mismatches, incorrect gamma settings, camera noise and many other artifacts, creating a perfect replica of a real surface for an augmented image is so challenging that in the end visible seams between real and virtual surfaces are almost inevitable.

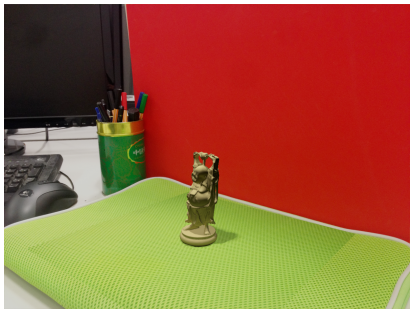
Often times, the influence an object exerts on reality is very localized and therefore AR simulations only model a patch of a surface beneath the object. Even with a very close match, the realism of the augmentation falls apart due to the visible discrepancy between reconstruction and reality (see Figure 3.1(a) and 3.1(b)).

Two key publications to solve this problem have been presented: The foundation by Fournier et al. [FGR92] was later popularized by Paul Debevec [Deb98]. Instead of rendering an impostor in place of the original object, the author suggests to extract the difference between an augmented and a regular impostor for the final rendering. In this manner, the augmented surface is minimally distorted by an inaccurate reconstruction.

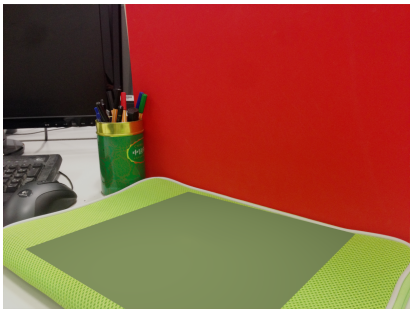
The idea follows a simple pattern: Suppose the background image to be



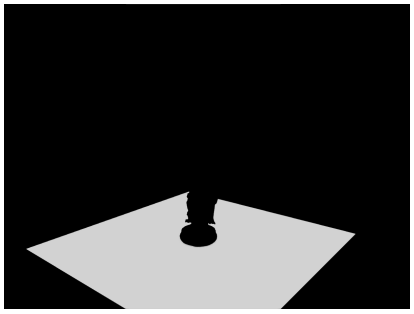
(a) Albedo mismatch



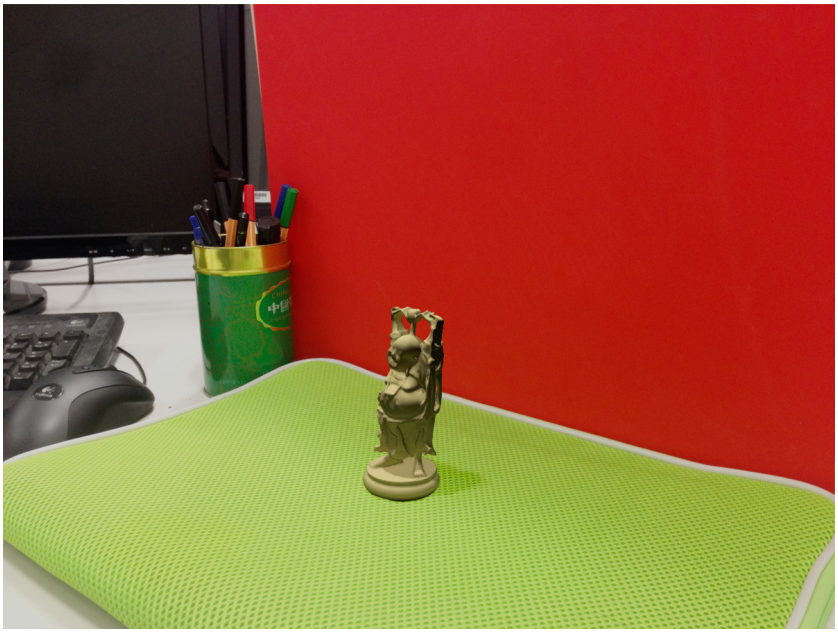
(b) Meso structure mismatch



(c) Without object



(d) Mask



(e) Extracted and augmented difference

Figure 3.1: Differential Rendering overview. (a) Visible discrepancy between reconstructed surface and the original. (b) Even small-scale differences to the augmented surface can create distracting seams. A solution to this problem is to augment only the extracted differential between (a) and (c) masked by (d).

augmented is called L_b. A local scene reconstruction is used to recreate a rendition L_μ of the scene as it exists without any modifications or augmentations. The discrepancy between the imperfect reconstruction L_μ and the real scene L_b is therefore:

$$\Delta L = L_\mu - L_b \tag{3.2}$$

The reconstructed scene is rendered a second time with an additional object augmenting it and the result is L_ρ. Since we now know the error between reconstruction and reality ΔL we can likewise compensate for it when augmenting the image using Equation (3.2) :

$$\begin{aligned} L_f &= L_\rho - \Delta L && (3.3) \\ &= L_b + (L_\rho - L_\mu) && (3.4) \end{aligned}$$

The term $(L_\rho - L_\mu)$ is called the *differential* between augmented and unaugmented scene. This extraction can be simply added on top of the background image L_b to compose the final image L_f. An overview is shown in Figure 3.1.

The popularity of Differential Rendering among Augmented Reality renderers (see for instance [GM00, BGWK03]) can be attributed to its simplicity: With a masking function any rendering algorithm can be modified to augment existing scenes with new elements, with minimal impact due to inaccurate reconstructions. However, this simplicity comes at a cost with a significant impact once computationally expensive rendering algorithms are introduced to a real-time rendering environment for AR: All computations have to be done *twice*. Especially in the presence of GI computations or participating media, Differential Rendering may become unfeasible, independent of the method chosen to compute such effects.

Based on the following three observations optimizations for this behavior can be derived:

1. **High coherency** Images L_ρ and L_μ are often highly correlated. Surfaces may appear different due to changes in illumination caused by the introduction of a new object into the scene (for instance shadowed regions), but they do not change their material properties. The expensive computation of the integral in Equation (2.7) can be avoided if the initial real light propagates only the necessary change in illumination instead of a change in appearance or reflection.

2. **Localized influence** Only a local surrounding around the object is usually influenced by shadows or indirect bounces. Computations can be reduced by defining a tight boundary of influence around the augmenting object to avoid unnecessary, dual shading of unaffected regions.

3. **View independence** Differential influence on the real scene may exhibit, under static conditions, view independence. Diffuse surfaces darkened by blocked light sources from a virtual object, as well as the diffusely scattered indirect bounces from a virtual object, are independent of the observers position and can be computed in one block instead of a per-frame differential. By decoupling view-independent illumination differences from the image, one computation can be used for changing views.

3.1.2 Contribution

Following these observations I propose to create a new view on relighting by modeling only the propagated change or *delta* forced onto a field of visible energy [Fra13b, Fra13a].

I present the notation of the delta operator, which extracts the difference in illumination of an an augmented scene at each bounce instead of the camera sensor at the very end. This delta operator is used to compute the *Delta Radiance Field* L_Δ, which represents a radiance field of change and

can be thought of as a layer on an existing field. Theoretical observations are presented, which will yield concrete results in the next two chapters. Global illumination methods adapted to this operator will benefit from reduced computation overhead.

3.2 Formal Definition

The five dimensional plenoptic function $L(\mathbf{x}, \vec{\omega_o})$ can be evaluated for any surface point $\mathbf{x}$ in any direction $\vec{\omega_o}$.

$$\begin{aligned} L(\mathbf{x}, \vec{\omega_o}) &= L_e(\mathbf{x}, \vec{\omega_o}) + & (3.5) \\ & \int_{S^2} L_i(\mathbf{x}, -\vec{\omega_i}) f\left(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}\right) \left\langle \vec{n_x}, \vec{\omega_i} \right\rangle_+ d\vec{\omega_i} & (3.6) \end{aligned}$$

It is composed from these parts: L_e is the emissive radiance; f is a BSDF; $\left\langle \vec{n_\mathbf{x}}, \vec{\omega_i} \right\rangle_+$ is the geometric term for normal $\vec{n}_\mathbf{x}$ at $\mathbf{x}$ and incident light direction $\vec{\omega_i}$. We can expand the equation into a Neumann series with the integral expressed as a linear transport operator $\mathbf{T}$. Operator $\mathbf{T}$ can be furthermore split up into two components: A *reflection* operator $\mathbf{K}$ and a *propagation* operator $\mathbf{G}$.

$$\mathbf{T} = \mathbf{KG} \tag{3.7}$$

$\mathbf{K}$ represents the surface shading operation applied to incident light, and $\mathbf{G}$ the propagation of light, for instance through a raytracing operation. If $\mathbf{T}$ is contractive (i.e., $||\mathbf{T}|| \leq 1$)[1], then L has a solution:

[1] A necessary condition to ensure energy conservation.

$$
\begin{aligned}
L &= L_e + \mathbf{T}L && (3.8)\\
(\mathbf{I}-\mathbf{T})L &= L_e && (3.9)\\
L &= (\mathbf{I}-\mathbf{T})^{-1}L_e = \sum_{i=0}^{\infty}\mathbf{T}^i L_e && (3.10)
\end{aligned}
$$

Each term $\mathbf{T}^i$ intuitively represents propagation and scattering of one bounce of light, where $L_0 = L_e$ is the emissive light only, $L_1 = \mathbf{T}L_e$ is direct light, $L_2 = \mathbf{T}^2 L_e$ the first indirect bounce of light, and so on.

Consider an existing radiance field L_μ. When we insert another object O into the scene covered by L_μ, the properties of the radiance field change: Light is either blocked or scattered by the newly inserted geometry, or it may even add energy due to emissive surfaces. To account for this *change* in the radiance field, consider another field L_ρ which has the same light configuration as L_μ and additionally contains O. The change in a radiance field L_μ by introducing O is therefore a *Delta Radiance Field* $L_\Delta = L_\rho - L_\mu$. By expanding the equation into its series components it is evident that a new operator can be created to propagate the change extracted from the initial lighting conditions L_e:

$$
\begin{aligned}
L_\Delta &= L_\rho - L_\mu && (3.11)\\
&= \sum_{i=0}^{\infty}\mathbf{T}_\rho^i L_e - \sum_{i=0}^{\infty}\mathbf{T}_\mu^i L_e = \sum_{i=0}^{\infty}\left[\mathbf{T}_\rho^i L_e - \mathbf{T}_\mu^i L_e\right] && (3.12)\\
&= \sum_{i=0}^{\infty}\mathbf{T}_{\Delta_i} L_e && (3.13)
\end{aligned}
$$

The term $\mathbf{T}_{\Delta_i}$ is called *delta transfer operator* and includes both positive bounces from a virtual object back onto reality as well as a correction term for now occluded transport.

3.3 Observations

To create a consistent and plausible image, operator $\mathbf{T}_{\Delta_i}$ must not violate the conditions already in place for a regular transport operator $\mathbf{T}$. In essence, all energy L_e which is propagated forth from the initial emissive surfaces is now scattered on newly inserted interfaces which change the IOR at some point in space. To maintain energy conservation, the original propagation along the original path before the change needs to be reversed so that the redistribution of energy does not change the total amount present in the scene, i.e. $||\mathbf{T}_{\Delta_i}|| = 0$. From this definition it is apparent that operator $\mathbf{T}_{\Delta_i}$ not only forces paths to deviate from their original trajectory, but must also propagate a negative quantity of energy called *antiradiance* along their original direction to compensate for excess energy.

To correctly support negative transport through translucent media or other subsurface scattering phenomena, antiradiance needs to be inserted at the position of the new scattering event. A special case for surface radiance was presented by Grosch [Gro05] where negative photon flux is inserted into the photon map as compensation.

Antiradiance $A(\mathbf{x}, \vec{\omega})$ is computed at the first intersection of a real light bounce with a virtual interface:

$$A(\mathbf{x}, \vec{\omega_o}) = -L_\mu(\mathbf{x}, -\vec{\omega_i}) \tag{3.14}$$

From there on it propagates like regular light does, canceling out excess radiance in regions which no longer receive light due to new objects in the scene. This propagation occurs without further regard to the virtual object, only interacting with real surfaces. This way all excess energy in the scene is removed, while new energy is scattered according to the new geometric condition of the scene. A visual representation of this behavior can be seen in Figure 3.2.

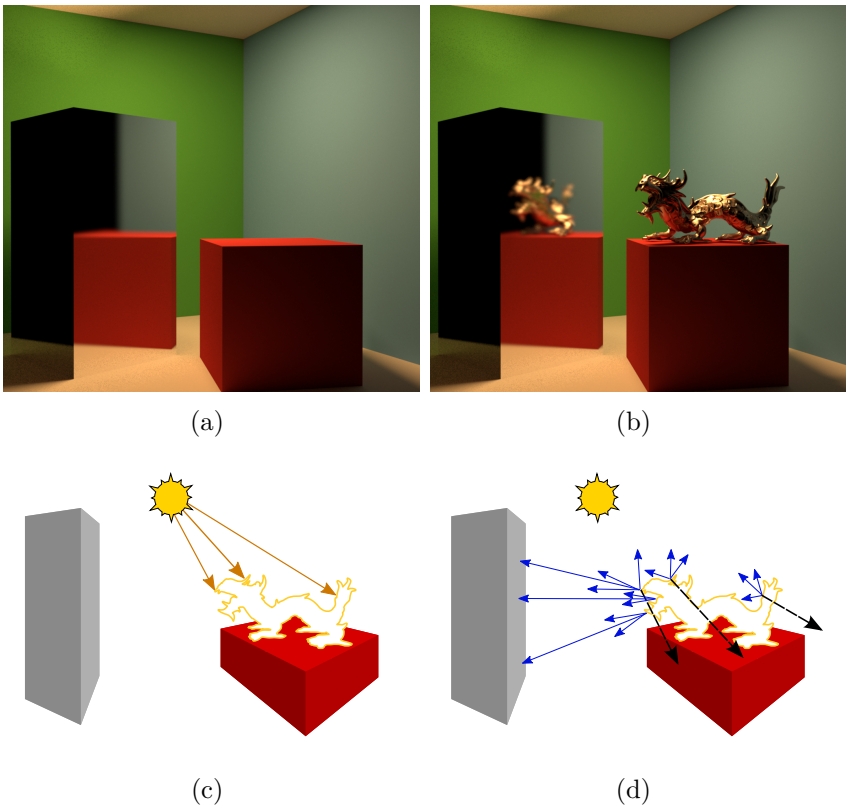


Figure 3.2: Delta propagation. An empty scene in (a) is augmented by the dragon figure (b). After the first path segment hits the surface of the augmenting object (c) light is split into antiradiance and regular scattered light (d). Antiradiance (black arrows) at the new interface propagates as if the interface is not present. Scattered light (blue arrows) behaves regularly and bounces between both the scene and the virtual surfaces.

The amount of all scattered light off new interfaces is less or equal to the amount of antiradiance to conserve all energy in the scene.

3.4 Implementation

A delta transfer operator can be integrated into the path tracing framework. By keeping track of all virtual surfaces, a special class of antiradiance paths[2] can be be traced through the real scene, while scattered light is distributed

[2]Not to be confused with simple *shadow rays* or similar often identified in various relighting publications.

with the regular framework. For methods with caching operations such as Radiance Caching, Irradiance Caching, Instant Radiosity, Photon Mapping or others it is important to distinguish between negative and positive hits as long as the propagation continues. As soon as both quantities are overlapping, it is no longer possible to distinguish antiradiance paths from regular ones.

Operator $\mathbf{T}_{\Delta_i}$ furthermore can be expanded for all rendering algorithms, which is an attractive quality for real-time rendering. To model direct and indirect change of n bounces in a radiance field, operators $\mathbf{T}_{\Delta_i}$ can be applied to incoming light to calculate a new radiance field, which can then be added onto L_μ to approximate L_ρ.

$$L^\rho \approx \sum_{i=0}^{n} \left(\mathbf{T}_\mu^i L_e + \mathbf{T}_{\Delta_i} L_e \right) \tag{3.15}$$

The basic setup used for the algorithms in the following chapters is depicted in Figure 3.3: The scene is captured using a Microsoft Kinect, which provides an RGB-D image. A secondary fish-eye camera captures incident distant lighting. This camera is used to reconstruct point light sources or to provide an irradiance map for shading purposes. The scene is geometrically registered using a simple marker-based approach. Any other initialization or tracking technique may be used here as well. In this thesis, the approach is realized with OpenCV [Bra00].

3.5 Conclusion

In this chapter, I presented a mathematical framework to model change in a radiance field. Building algorithms according to a delta transfer operator will lead to more efficient global illumination relighting methods for applications such as Augmented Reality, because the relighting solution is no longer required to calculate two GI solutions to extract the illumination differences, cutting previous costs in half. Shifting the extraction of the difference be-

Figure 3.3: A relighting setup: The scene is recorded and partially reconstructed with a Kinect sensor. A marker is used to geometrically register the scene and a secondary UEye UI-2230-C fish-eye lens camera pointing upwards is used to capture the incident illumination near the augmenting object.

tween real and virtual into the operators responsible for calculating bounces has performance benefits. In the following two chapters I will first expand and solve **Problem 1** (the shading of virtual objects) discussed in Section 1.1, and then to present several implementations of $\mathbf{T}_{\Delta_i}$ for relighting algorithms which solve **Problem 2**. By exploiting the insights of this chapter, formerly expensive global illumination methods are now available for both tasks, which addresses **Problem 3**.

Chapter 4

Shading Virtual Surfaces

4.1 Introduction

The first objective after geometric registration in Augmented Reality is to shade the virtual object according to real incident light. To believably aug-

ment a real scene with an additional object, this object needs to appear consistent with real counterparts. Changes in the surrounding real lighting conditions need to be reflected on the virtual object, just as they would on any other real object. Furthermore, the inter-reflections on the surface of the object and the inter-reflections between real and virtual surfaces need to be accounted for as well. Without these global effects, augmenting objects tend to look out of place and are immediately recognized as objects not being part of the real context.

In this chapter I will formulate two shading systems for virtual objects, both of which account for natural illumination (i.e., light derived from a captured surrounding instead of modeled point lights) and are able to compute reflected light on a wide variety of materials, accounting for smooth inter-reflections in real-time. These systems address different requirements: The simulation of objects with rigid geometry and static surface properties, and the simulation of objects which have neither.

4.1.1 Related Work

Debevec [Deb98] captures real light from an HDR light-probe instead of using reconstructed point light sources. The resulting images are mapped onto objects to simulate the reflection of real light on virtual objects. Differential rendering is introduced to superimpose lighting interaction between virtual and real surfaces on a background image, which has been subsequently turned into an interactive method [GM00] ignoring however any kind of inter-reflection between objects.

Karsch et al. [KHFH11, KSH+14] describe a method to insert virtual objects into legacy photographs. With the help of user annotations to define occluding geometry and light sources or light shafts or similar pre-learned environments, scene geometry and lighting is estimated and several reconstruction processes are avoided. A similar approach with environment maps can estimate the real light configuration better [BCD+13]. In contrast, the methods in this dissertation focus on real-time insertion into dynamic scenes instead of static images. Light sources and geometry are automatically re-

constructed from camera images capturing scene depth, incident environment light or tracked light sources.

Screen Space Direct Occlusion (SSDO), a method introduced in [RGS09] that is similar to Screen Space Ambient Occlusion (SSAO), samples the neighboring color buffer of a texel to gather near-field indirect illumination. While this method is very flexible, it is prone to the same limitations as SSAO.

Many other methods are available to extract and map incident real light onto virtual surfaces (see Section 2.3.3.2). All of them however can be categorized into either image-based lighting methods which reconstruct individual light sources or an analytical function of light distribution across the hemisphere, or methods which use sensors to detect light sources.

4.1.2 Contribution

In this chapter, I explore two systems to capture real light and shade a virtual surface using combinations of different methods. The general idea for shading virtual objects involves to split up the rendering equations integral, which convolves incident light with the transport operator, into two segments concerned with determining low-frequency $\mathbf{T}_d$ and high-frequency $\mathbf{T}_s$ transport independently. This move is motivated by the fact that low-frequency transfer such as diffuse reflection and ambient occlusion can be evaluated with efficient filtering methods using low-frequency basis functions, whereas high-frequency signals cannot be processed and mapped as efficiently by the same methods.

$$
\begin{aligned}
L\left(\mathbf{x}, \vec{\omega_o}\right) &= L_e\left(\mathbf{x}, \vec{\omega_o}\right) + \int_\Omega f_r\left(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}\right) L_i\left(\mathbf{x}, -\vec{\omega_i}\right) \left\langle \vec{n_\mathbf{x}}, \vec{\omega_i} \right\rangle_+ d\vec{\omega_i} && (4.1)\\
&= L_e\left(\mathbf{x}, \vec{\omega_o}\right) + && (4.2)\\
&\quad \int_\Omega \left(\mathbf{T}_d(\mathbf{x}, \vec{\omega_o}) + \mathbf{T}_s(\mathbf{x}, \vec{\omega_o}, \vec{\omega_i})\right) L_i\left(\mathbf{x}, -\vec{\omega_i}\right) d\vec{\omega_i} && (4.3)
\end{aligned}
$$

A wide array of shading methods from the scientific literature can be analyzed and used for particular effects, however preconditions for the virtual object such as rigidness are important factors to select a specific combination. I propose two combinations for two different use cases, and analysis of each combination of methods reveals the particular weaknesses and strengths. The two conditions for virtual objects addressed in this chapter are:

- Dynamic transfer for AR object shading: A combination of irradiance mapping, specular importance sampling and visibility approximation using ambient occlusion [FJ08b]

- Precomputed transfer for AR object shading: A combination of PRT for smooth visibility and Spherical Gaussian encoded materials [FJ08a, FF12]

Both methods are tested for simulating different appearances and their efficiency. A final discussion will further extend both methods to simulate reflection of other virtual objects.

4.2 Shading of Dynamic Objects

In AR environments, changes in the surrounding illumination and geometry can happen any time. This may or may not be true for the virtual object. However, in an AR environment which for instance presents a configurable car, changing materials or geometric relations must be anticipated by the renderer. In this section, I will formulate an efficient system to render a virtual object under varying natural illumination with no preconditions for either materials or rigidness.

4.2.1 Image Based Lighting

Instead of parametric descriptions of light sources, a system can be imagined which derives lighting information from images modeling the surrounding environment, for instance in a panoramic view. This process can either create new parametric light sources or represent light as an analytical function over a domain such as the hemisphere of a point. If this function is merely an index into the image, it can be thought of as a form of environment mapping [BN76b], where the appearance is computed by calculating reflection vectors to index into a spherical image of an infinitely far away surrounding. Even though the parametrization of the reflecting surface may change the resulting reflection often times looks plausible.

4.2.1.1 Irradiance Mapping

In this section I will briefly summarize Irradiance Mapping with spherical harmonics. Based on environment mapping several publications noted that an entire class of images can be imagined which represent different kinds of reflections of a surface. An environment map could be *pre-convolved* with a BRDF, for instance a simple purely diffuse Lambertian kernel, which when mapped onto the surface of an object creates the appearance of a diffuse reflection.

Ramamoorthi and Hanrahan [RH01] have demonstrated that a reconstruction of diffuse irradiance from an irradiance map with an average error margin under 3% can be achieved with only 9 coefficients in the spherical harmonic basis. This method can be used for pre-filtering, generating a diffuse irradiance map.

The associated Legendre polynomial is defined as follows:

$$\frac{d}{dx}\left[(1-x^2)\frac{dy}{dx}\right] + \left[l(l+1) - \frac{m^2}{1-x^2}\right] y = 0 \tag{4.4}$$

The polynomial $P_l^m(x), x \in [-1, 1]$ comes with two parameters: $l \in \mathbb{N}$ is a *band index* and $m \in [0, l]$ is an identifier of a orthogonal function within this band. Each band spans its own basis, and successive bands provide coverage of higher frequencies. Because numerical evaluation of Equation (4.4) is difficult, a recursive relation of three rules can be used instead:

1. $(l-m)P_l^m = x(2l-1)P_{l-1}^m - (l+m-1)P_{l-2}^m$
2. $P_m^m = (-1)^m(2m-1)!!(1-x^2)^{m/2}$
3. $P_{m+1}^m = x(2x+1)P_m^m$

This 1D polynomial can used to construct the spherical harmonic function $Y_l^m(\theta, \phi)$ which is generally defined on imaginary numbers. For the purpose of approximating real valued functions on a sphere it is convenient to use the the *real spherical harmonic function*, which is defined as follows:

$$y_l^m(\theta, \phi) = \begin{cases} \sqrt{2}K_l^m \cos(m\phi)P_l^m(\cos\theta), & m > 0 \\ \sqrt{2}K_l^m \sin(-m\phi)P_l^{-m}(\cos\theta), & m < 0 \\ K_l^0 P_l^0(\cos\theta), & m = 0 \end{cases} \tag{4.5}$$

In this definition, K_l^m is a normalization factor:

$$K_l^m = \sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}} \tag{4.6}$$

A linearized version $y_l^m(\theta, \phi) = y_i(\theta, \phi)$ of the coefficients can be created with the following relation:

$$i = l(l+1) + m \tag{4.7}$$

An efficient implementation of Equation (4.5) can be found in [PTVF07].

4.2.1.2 Diffuse

We use Monte Carlo estimation to convolve a spherical function $I(\theta, \phi)$, which represents irradiance from an environment map, with a spherical harmonics basis to compute coefficients c_i:

$$c_i = \int_\Omega I(\vec{\omega}) y_i(\vec{\omega}) d\vec{\omega} \tag{4.8}$$

The original signal can be reconstructed on the GPU with:

$$\overline{I}(\theta, \phi) = \sum_i^N c_i y_i(\theta, \phi) \tag{4.9}$$

By using just 9 coefficients (as [RH01]), we effectively reconstruct a low-frequency variant of the original input, which serves as diffuse irradiance and can be directly mapped onto the surface of a virtual object [FJ08b, JFDB07]. Highly specular reflection can be simulated with the help of the original environment map, however naive blending between both does not result in a glossy appearance (see Figure 4.1).

4.2.1.3 Filtered Specular Importance Sampling

With the help of the previous result we can effectively simulate diffuse and highly specular reflections off a virtual object. To include physically-based

(a) (b) (c)

Figure 4.1: Varying surface roughness of a virtual object. (a) Diffuse reflectance is simulated with the help of a low-frequency irradiance map. (b) Glossy reflection through blending a low-frequency irradiance map with the original reflection used in (c) map leads to an unwanted layered material appearance.

glossy reflections off the surface of a virtual object from real environment light, another strategy has to be chosen. With unknown, possibly changing surface roughness, pre-convolution of specular BRDFs with arbitrary roughness quickly leads to a huge database of only slightly differing environment maps. A naive approach to tackle this problem is to generate a stack of pre-convolved maps from which all intermediate states are interpolated, usually accelerated by hardware mipmapping, which leads to considerable bias as the parametrization of a environment map is unfit to filter on a spherical domain. Instead, one can employ importance sampling techniques combined with a defined small set of pre-convolved environment maps with a fixed kernel [KC08].

Importance sampling replaces a random distribution across a hemisphere with an important distribution of directions (s_θ, s_ϕ) which leads to a quicker convergence when convolving incident light and the specular function f_s. When convolving lighting with a surface BRDF, many samples may be needed to ensure a noise-free, temporally coherent result. To reduce this number, samples may be weighted according to their relative *importance* in the integration result. As seen in the Section 2.1.4.2, an integral can be determined with a Monte Carlo estimator:

$$\langle I \rangle = \frac{1}{N} \sum_{i=0}^{N} \frac{f_s(x_i)}{p(x_i)} \tag{4.10}$$

The relative weight for each sample $f_s(x_i)$ is determined by a Probability Distribution Function (PDF) $p(x_i)$. For unknown illumination it may be difficult to create a PDF. Another guide which can be used to create better samples is the specular roughness, more specifically the Normal Distribution Function of the specular component f_s in the BRDF. Since specular reflection is sampled inside a lobe of the hemisphere, samples can be distributed accordingly. In this work, I chose the popular Trowbridge-Reitz/GGX [TR75, WMLT07] distribution to model specular roughness.

The Normal Distribution Function D of a GGX distribution is defined as follows:

$$D\left(\vec{m}\right) = \frac{\alpha^2}{\pi\left(\cos^2\left\langle\vec{n},\vec{m}\right\rangle_+\left(\alpha^2-1\right)+1\right)^2} \tag{4.11}$$

The NDF serves as PDF from which one derives samples with hemispherical direction (θ, ϕ):

$$p(\theta,\phi) = \frac{\alpha^2}{\pi\left(\cos^2\theta\left(\alpha^2-1\right)+1\right)^2}\cos\theta\sin\theta \tag{4.12}$$

To generate samples according to either hemispherical direction, one first need to create two independent functions for θ and ϕ. For $p(\theta)$ one integrates along the domain of ϕ to generate a *marginal density function*:

$$p(\theta) = \int_0^{2\pi} p(\theta,\phi)d\phi = \frac{2\alpha^2}{\left(\cos^2\theta\left(\alpha^2-1\right)+1\right)^2}\cos\theta\sin\theta \tag{4.13}$$

Using the result, one can create an independent conditional density function $p(\phi)$:

$$p(\phi) = p(\phi|\theta) = \frac{p(\theta,\phi)}{p(\theta)} = \frac{1}{2\pi} \tag{4.14}$$

For all isotropic NDFs $p(\phi)$ is identical. Integrating both functions along their respective domain yields a Cumulative Density Function (CDF) for each. The CDF for $p(\phi)$ is trivial:

$$P(s_\phi) = \int_0^{s_\phi} p(\phi)d\phi = \int_0^{s_\phi} \frac{1}{2\pi}d\phi = \frac{s_\phi}{2\pi} \tag{4.15}$$

To gain a sample direction s_ϕ, we set $P(s_\phi)$ to a uniform random variable ξ_ϕ and solve for s_ϕ:

$$P(s_\phi) = \frac{s_\phi}{2\pi} = \xi_\phi \tag{4.16}$$

$$s_\phi = 2\pi\xi_\phi \tag{4.17}$$

Likewise, by integrating $p(\theta)$ along θ one constructs a CDF $P(s_\theta)$:

$$P(s_\theta) = \int_0^{s_\theta} p(\theta)d\theta = \tag{4.18}$$

$$2\alpha^2\left(\frac{1}{(2\alpha^4 - 4\alpha^2 + 2)\cos^2 s_\theta + 2\alpha^2 - 2} - \frac{1}{2\alpha^4 - 2\alpha^2}\right) \tag{4.19}$$

Setting $P(s_\theta)$ to another uniform random variable ξ_θ, important sample directions s_θ are:

$$P(s_\theta) = \xi_\theta \tag{4.20}$$

$$s_\theta = cos^{-1}\left(\sqrt{\frac{1-\xi_\theta}{(\alpha^2-1)\,\xi_\theta+1}}\right) \tag{4.21}$$

However, even with a sampling pattern now guided by the specular component too many samples are needed to bring down the variance to a reasonable limit. Hardware accelerated filtering can be used in conjunction to reduce the amount of texture fetches on the environment map further. Instead of fetching multiple samples, the appropriate mipmap level is determined from a given specular roughness to fetch their prefiltered result. To do this, a level l can be extracted from the normal density p with:

$$p = D\left(\left\langle \vec{n},\vec{h}\right\rangle_+\right)\frac{\left\langle \vec{n},\vec{h}\right\rangle_+}{\left\langle \vec{v},\vec{h}\right\rangle_+} \tag{4.22}$$

$$l = \max\left(0,\frac{1}{2}log_2\left(\frac{w^2h^2}{N}-log_2(p)\right)\right) \tag{4.23}$$

To avoid under-sampling, a factor $\frac{1}{2}$ is multiplied in l to adhere to a Nyquist frequency. In Figure 4.2, the spherical augmenting object (known as the *Mitsuba sphere*) is composed out of three parts with different materials: A red metallic paint for the outer shell, a shiny golden inner sphere and a copper casing. Each material is simulated with a measured Fresnel value and each one uses a different roughness value α. Combined with the diffuse irradiance gathered in the previous section, a full BSDF can be simulated.

4.2.1.4 Visibility

Image-based lighting can be used to map real light onto virtual objects. Irradiance mapping and environment mapping however do not handle local occlusions (i.e., self-occlusion from the virtual object itself). Visibility queries

Figure 4.2: Augmenting Mitsuba sphere with three different metallic surfaces of varying roughness.

are expensive and may be replaced by low-frequency ambient occlusion. In [FJ08b] we have used hardware occlusion queries to estimate the AO factor for each vertex, which have since been exchanged with SSAO. Figure 4.3 shows the augmenting Ajax bust with mapped AO which is later combined with the the incident light computed from the previous sections. In the case of glossy or highly specular surfaces, Kozlowski and Kautz [KK07] have found that perceived realism is barely affected by this approximation.

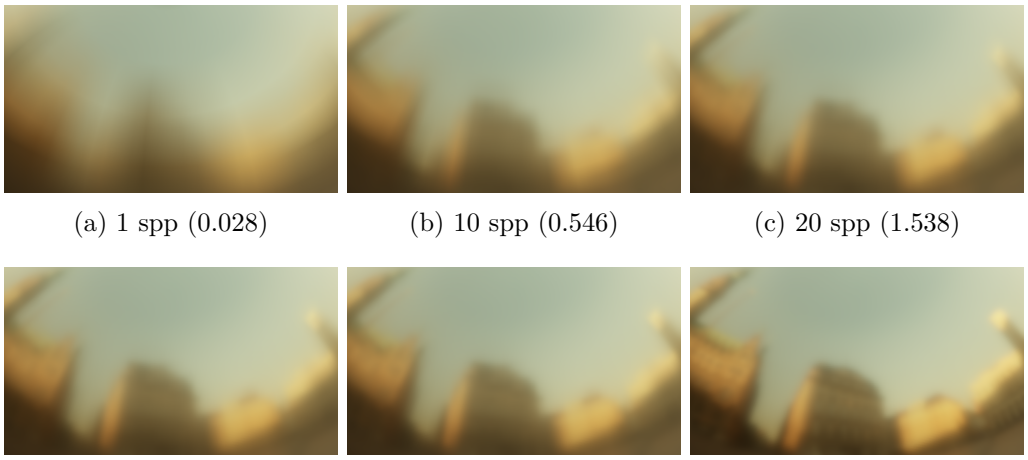
4.2.2 Results

I have shown an enhanced method of our previous publication in [FJ08b, JFDB07] to render augmenting objects under natural illumination. In Figure 4.4, the augmenting Stanford Dragon's surface is lit from diffuse light evaluated with a low-frequency irradiance map. SSAO adds visibility, and specular reflections are evaluated using Filtered Importance Sampling using a GGX NDF at 40 samples per pixel (spp).

The time to render a frame varies mostly with the amount of specular samples

Figure 4.3: Visibility approximation for dynamic scenes. SSAO is used to augment incident light with self-occlusion.

Figure 4.4: Stanford Dragon augmenting a background image. The dragon's surface is shaded with dynamic ambient occlusion.



(a) 1 spp (0.028) (b) 10 spp (0.546) (c) 20 spp (1.538)

(d) 30 spp (2.328) (e) 40 spp (3.162) (f) Reference

Figure 4.5: Filtered Specular Importance Sampling. This overview shows the number of samples per pixel, the evaluation in milliseconds and the resulting image quality of a spherical glossy reflection rendered at 720p.

collected to render glossy reflections. In Figure 4.5 an overview shows the difference in quality when using different amount of samples.

Highly glossy surfaces can produce aliasing with low sample counts, exposing the sampling pattern on the surface. This can be fixed with higher sampling counts, albeit at higher rendering cost. Using SSAO as visibility substitute in ambient lit surroundings is a reasonable choice, however when natural illumination exhibits a strong directional component as in Figure 4.1 the assumption of ambient occlusion breaks.

4.3 Shading of Rigid Objects

Augmenting objects are often rigid and do not exhibit changing surface properties, such as is the case when displaying cultural artifacts like busts, tablets or other artifacts. This knowledge can be exploited to pre-compute surface and subsurface transfer. For diffuse reflection, the shortcomings of non-local transport in the previous sections can be overcome for rigid objects by employing Precomputed Radiance Transfer. While we can certainly precompute

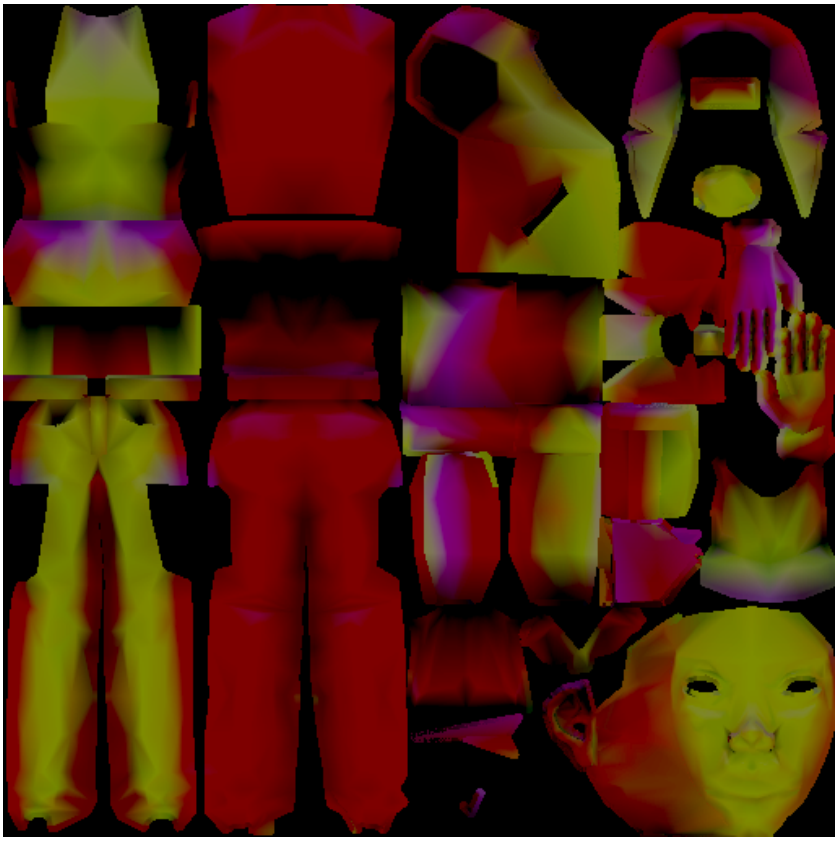


Figure 4.6: A PRT texture of the *Tiny* model from the Microsoft DirectX SDK. The colors represent the first three spherical harmonic coefficients per pixel.

high-quality ambient occlusion, PRT allows to incorporate directionally dependent effects as well at only slightly higher storage cost. In this section, I will formulate an efficient system to shade rigid objects with static surface properties.

4.3.1 Diffuse Precomputed Radiance Transfer

In a pre-process, coefficients t_c are determined for each surface point. In practical terms, we can sample a mesh either at the level of each vertex and store these coefficients as vertex attributes, or we can likewise sample a parametrized surface and store coefficients in multiple textures [FF12]. Figure 4.6 shows one of a set of textures encoding PRT coefficients in the spherical harmonics base.

The basic form of PRT presented in Section 2.2.1.3 must be adapted to support view-dependent effects to Matrix Radiance Transfer [LK03]. Instead of a coefficient vector, each surface point saves a representation of the BSDF as a *matrix* **B** to convert incident radiance to exit radiance.

$$\tilde{L}_p\left(\mathbf{x}, \vec{\omega_o}\right) = \sum_i \left(\sum_j B_{ij}(\mathbf{x}) l_j \right) \Phi_i(\vec{\omega_o}) \tag{4.24}$$

The fidelity of a view-dependent effect is furthermore coupled to the basis function it was encoded in. Here, the spherical harmonic basis has one major drawback: High frequency details cannot be efficiently encoded in a set of few coefficients. Several basis functions have been proposed which can more adequately capture high-frequency details such as Haar-Wavelets, Spherical Wavelets, Spherical Radial Basis, Spherical Piecewise Basis etc. Independent of the basis function though, matrix radiance transfer consumes $O(n^2)$ operations per surface point and an equal amount of coefficients have to be saved to represent a BSDF. To avoid this massive overhead we can consider other basis functions in which glossy and specular reflections may be stored more efficiently and are evaluated more quickly.

4.3.2 Specularity via Gaussians

A method in [GKD07a] approximates an isotropic BRDF as a Sum of Gaussian (SoG) functions. The rendering equation integral is represented as a dot product of two functions with direction $\vec{\omega_i}$ being replaced with a spherical displacement $\vec{\delta_i}$ to reflection direction $\vec{\omega_r}$:

$$L(\mathbf{x}, \vec{\omega_o}) = \int_\Omega \mathbf{T}_{\mathbf{x},\theta_o}(\vec{\delta_i}) L_i(\vec{\omega_r} + \vec{\delta_i}) d\vec{\delta_i} \tag{4.25}$$

Let $\mathbf{T}_\mathbf{x}(\vec{\omega_i}, \vec{\omega_o}) = f_r\left(\mathbf{x}, \vec{\omega_i}, \vec{\omega_o}\right) \left\langle \vec{n_\mathbf{x}}, \vec{\omega_i} \right\rangle_+$ be a *transfer function* at surface lo-

cation $\mathbf{x}$ which represents the rendering integral apart from the incident light L_i. Approximating $\mathbf{T}_{\mathbf{x},\theta_o}$ as a sum of K Gaussian functions G for N different viewing angles θ_o yields $N \times K$ coefficients: The mean μ_k, standard deviation σ_k and a weight w_k.

$$\mathbf{T}_{\mathbf{x},\theta_o}(\vec{\delta_i}) \approx \tilde{\mathbf{T}}_{\mathbf{x},\theta_o}(\vec{\delta_i}) = \sum_{k}^{K} w_{k,\theta_o} G_{\sigma_{k,\theta_o}}(\mu_{k,\theta_o} - \vec{\delta_i}) \tag{4.26}$$

These coefficients can be saved in two small textures of height K and width N: A texture for all weights w_k for each Gaussian $k \in K$ of each view $n \in N$ as RGB values for each color channel of the BRDF response. Another texture stores the mean μ_k and the standard deviation σ_k per k per n. An example BRDF of a golden-metallic paint is illustrated in Figure 4.7. At runtime, a pre-filtered environment map is supplied, where each mipmap stores the same texture with a filter of σ^2 of the previous mipmap level. Even though not accurate, hardware filtering can be used to generate these mipmap levels automatically.

To compute the result a shader determines the coefficients from the pre-filtered environment map $(G * L_i)(\vec{\omega_r} + \mu_k)$, i.e., the convolution of a Gaussian function with incident light L_i, for any direction and compute the dot product of transfer coefficients and light. The Sum of Gaussians method can be used to augment low-frequency PRT with high frequency reflections. A result can be seen in Figure 4.7.

When augmenting objects into image streams, the combination of PRT and SoG is a flexible combination to precompute expensive low-frequency effects and still be able to use highly glossy surface lacquer or paints from measured data in very small containers. In Figure 4.8, a scene with a lamp has been set up showing interactivity with varying incident light on each row. The upper four images show a metallic paint from [GKD07a], and the lower four images show the same weights w_k with modified higher standard deviation σ_k, which is responsible for the rougher appearance.

(a) (b)

(c) (d)

Figure 4.7: Ajax bust rendered with a shiny golden material using the Sum of Gaussian method. In Figure (a) regular rendering is employed, while in Figure (b) visibility through spherical harmonic encoded PRT data is multiplied. The material's standard variance and means are encoded in (c), and the weights used can be seen in (d).

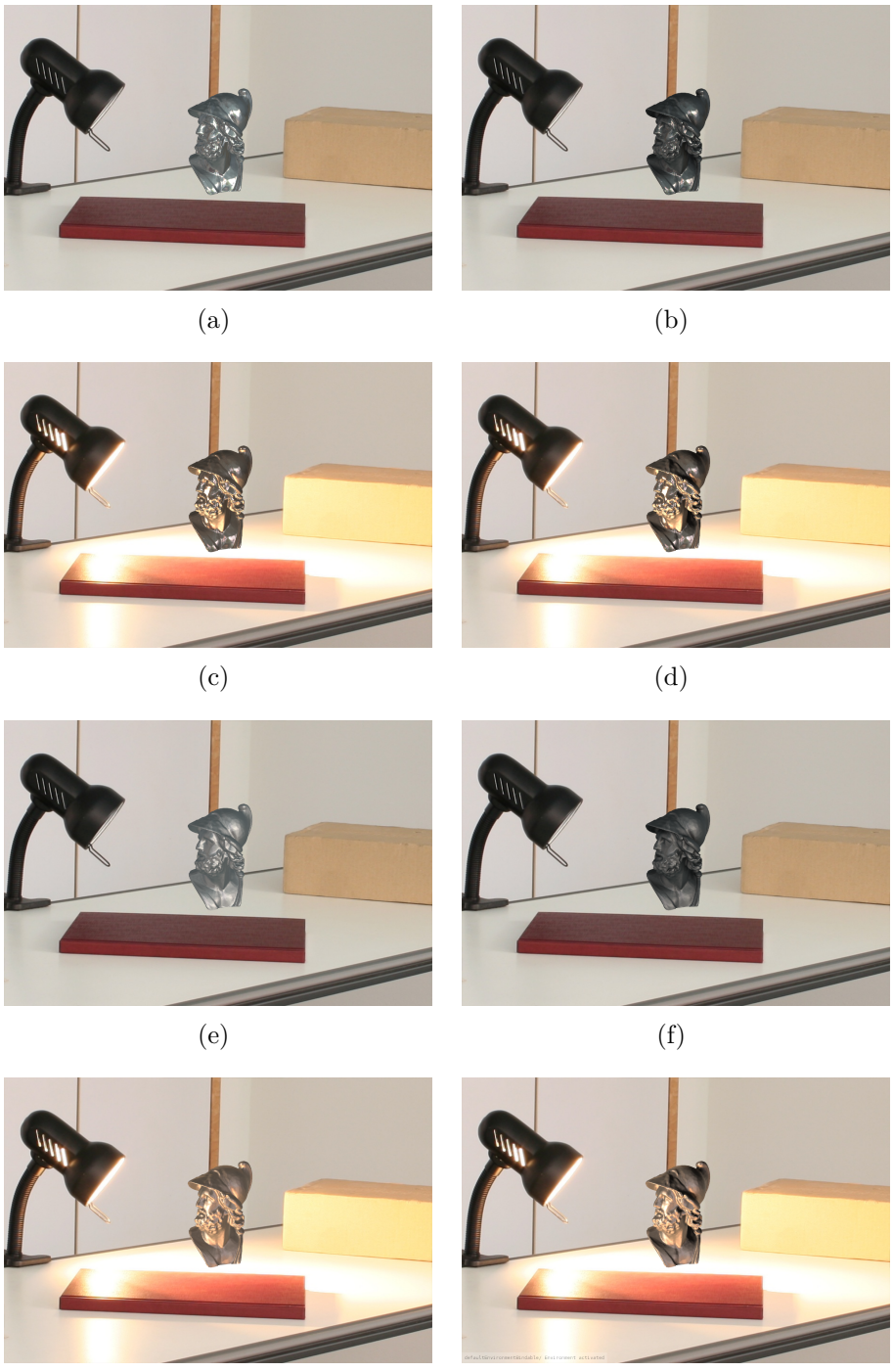
4.3.3 Other Material Bases

In [FF12] we review several practical basis functions. All of them can be used to shade a virtual object in a real environment, but there are some considerations to make.

Polynomial Texture Maps Malzbender et al. [MGW01] presents a texture format, the Polynomial Texture Map (PTM), which represents each pixels as a bi-quadratic polynomial to save the surface appearance under varying lighting conditions. A regular texture is appended with six coefficients per pixel per color channel, which not only cover bump map like effects, but because they are calculated from real BTF sets also capture non-local scattering effects, self-shadowing and indirect reflections. A sample can be seen in Figure 4.9.

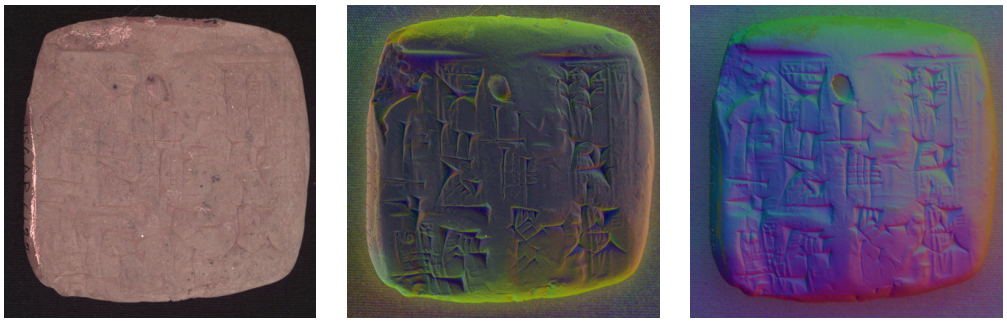
PTMs do not vary fundamentally from PRT[1]: A basis function is used to

[1]With the main difference of using point lights instead of functions when computing surface



(a) (b) (c) (d) (e) (f) (g) (h)

Figure 4.8: Augmenting Ajax bust. All images are rendered using a measured metallic paint from [GKD07a]. The left hand side shows rendering using Sum of Gaussians. The right hand side additionally handles visibility with 9 coefficients from spherical harmonics based PRT. Images (a-d) show a shiny and (e-h) a matte version of the paint.



(a) (b) (c)

Figure 4.9: An LRGB PTM of a cuneiform clay tablet is split into three components: (a) is the RGB albedo, followed by two images representing (b) the first and (c) the last three coefficients of the luminance polynomial. The PTM was created by HP Research Labs. The tablet belongs to the Archaeological Research Collection of the University of Southern California.

save coefficients for spatially dependent surface transfer. The basis function itself however is prone to suppress high-frequencies and is only suitable for monochromatic transfer.

Approximated BTF Approximated BTFs (ABTF) [KSS+04] are a subset of full BTF sets. Instead of supporting a fully bidirectional function, ABTF decouple the BRDF from meso-structures within the material. A stack of e.g. 15 images capture a material with a varying incident lighting angle θ. After the set has been captured, the average exit radiance of each image is calculated and all images are adjusted linearly in order to generate a stack of images lit at equidistant incidence angles (if they have not been captured this way already). The images are sorted by their average exit radiance and uploaded as 3D texture. At runtime, a shader linearly interpolates and indexes into this 3D texture with the geometric term $cos(\theta)$ to select the correctly lit pixel value of the material. The resulting pixel value is modulated with a regular material function such as the Blinn-Phong model. An environment map can be used to extract one or more light sources used for indexing into the 3D texture [Deb05].

reflectance.

Figure 4.10: Simulating appearance under natural illumination using PRT.

Although ABTFs can easily capture surface behavior for varying incident light, their small subset does not provide adequate flexibility to determine surface appearance under natural illumination.

4.3.4 Results

In [FJ08a, FF12] we show an application of this section: The *Ajax* bust is pre-processed with a raytracer, calculating self-occlusion multiplied by a Lambert factor coefficients per vertex and storing them in the SH basis. After capturing incident illumination and representing lower frequencies with SH, the dot product of all coefficients yields the appearance under natural illumination. In Figure 4.10, a small camera with a fish-eye lens hidden

beneath the augmenting virtual object captures a 180° map of the surrounding real light. The benefit over simple irradiance mapping is that surface transfer is now included and instead of ambient occlusion has a directional component.

By modifying PRT to compute only directional self-shadowing and instead calculate materials encoded as Sum of Gaussians we can extend the method to support arbitrary surface roughness. The result in Figure 4.8 shows that we can now produce different results using simple containers for measured materials. Since the evaluation of materials which are smooth or rough is not dependent on sampling any longer, but on a simple dot product, the shading cost remains constant independent of the type of reflection.

4.4 Discussion

Shading augmenting virtual objects comes down to a choice between relative flexibility and evaluation efficiency: If the object is static, i.e., it is for instance a stone bust, more complex surface appearance can be simulated at the cost of pre-computation necessary to prepare the object.

Based on the methods discussed in this chapter, we can derive two modes for shading either rigid or completely dynamic augmenting objects. In Figure 4.11, a real scene is augmented by the Ajax bust with a glossy surface exhibiting a characteristic silver Fresnel reflection. Both images show that nearly identical results can be produced with both methods presented in this chapter. The difference between the assumption of precomputed or completely dynamic behavior is reflected in the time to generate each frame, which for the dynamic image naturally comes at a higher yet overall low cost, and the subtle differences in shading due to the directional encoding of visibility.

To compare both methods with regards to efficiency, I have produced a test scene consisting of a sphere covering all pixels of a viewspace. All timings have been taken on an Intel i7 X980 and a NVIDIA GTX 780 graphics card.

(a) Dynamic evaluation

(b) Precomputed materials and visibility with dynamic illumination

Figure 4.11: Shading comparison of a rough silver metallic Ajax bust. (a) Completely dynamic shading with low-frequency irradiance mapping and Filtered Importance Sampling with a GGX specular NDF. (b) The same object with a directionally dependent visibility encoded in 16 SH coefficients and materials encoded as SoG. The subtle difference is noticeable at the back of the helmet, which features stronger shadowing of the dominant light source from the top.

In Table 4.1 both methods are associated to the respective resolution. The Precomputed variant delivers slightly better visual results at a fraction of the cost of the Dynamic version but cannot react to any smooth changes of either material properties or geometry changes (such as arbitrary surface deformations), as these are too expensive to recalculate per frame. The Dynamic computation is divided into the relatively expensive cost of the specular importance sampling (40 spp), the low-frequency irradiance mapping including the transformation, and SSAO. Here, the change in resolution affects the outcome much more severely. Both methods however perform within the boundaries of real-time behavior.

	480p	720p	1080p
Dynamic	1.292	6.093	14.407
Precomputed	0.02	0.021	0.04

Table 4.1: Comparison of timings taken for both methods with varying render context size. All entries are in milliseconds. In the Dynamic variant specular evaluation operates with 40spp.

Natural illumination through the use of environment mapping provides *distant* illumination and is only valid for the current position it is recorded from, i.e., the fish-eye lens camera's position. In Figure 4.10, this camera is located exactly beneath the virtual object (not visible in the image) and has to be moved with the virtual object in order to match the surrounding changes. While environment mapping itself is only an approximation to real reflections, parallax error (i.e., the offset between the fish-eye camera and the virtual object) is increasing noticeably with higher specularity. This error can be somewhat suppressed through warping techniques [SZ12] but will ultimately fail for nearby captured real objects. Another solution for largely static real scenery is to capture many different probes and interpolate between them (see for instance [GSHG98, GEM07]).

Another downside to pure natural illumination is that when simulating multiple virtual objects in a real context, they need to have multiple warped environment maps which however do not feature other virtual objects, i.e., the inter-reflection between virtual objects is missing. Modern real-time engines use several methods to determine specular reflections in objects, which successively fall back to another method if the current solution cannot provide any meaningful results [TVB+14]. The same model can be applied to incomplete reconstructions in AR simulations: Local reflections, i.e., reflections between virtual objects and nearby real reconstructed geometries, are computed with dynamic real-time GI algorithms and missing information is completed with the help of filtered lookups into the distant natural illumination. By combining these two methods a reasonable compromise between general surrounding illumination and important nearby reflectors can be found.

4.4.1 Light Propagation Volumes

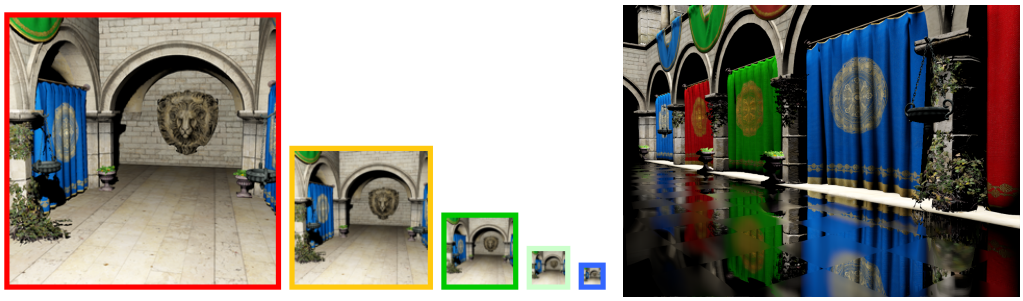
In [Fra13a] I have proposed to extend Light Propagation Volumes (LPV) [KD10] as a suitable implementation of a Delta Radiance Field. Each virtual object is encased in a small volume twice the size of its largest bounding box axis of 32^3 voxels. Indirect bounces from nearby objects are injected into the volume and then propagated through it. Their directional contribution is preserved as a low-frequency spherical harmonic encoding. After the propagation, the shading function simply queries the the volume for each surface point with the surface normal to derive the current indirect irradiance at this point.

Further discussion of LPV details is deferred to Section 5.3. LPVs can be used to quickly compute diffuse indirect bounces for each surface, but fail to capture glossy or specular details.

4.4.2 Screen Space Cone Tracing

In [HF14] we present a filtered sampling technique to compute glossy and highly specular Screen Space Reflections (SSR). SSR employs screen space ray-casting or ray-marching to find nearby intersections within the visible space. A ray $\vec{r}$ from a starting position $\mathbf{p}$ is followed until it either leaves the visible screen space or its current z-component has a depth larger than a pixel in the corresponding geometry buffer at the same position. An efficient algorithm can be constructed which increases the step-size along the ray with each step and once a *cross-over* is detected (i.e., the depth of the buffer has been passed), the direction is reversed and followed by half the amount of the current step-size. This method works very well for highly specular surfaces, but once glossy surfaces are introduced, many samples are needed to compute the appropriate response of a rough appearance. The expenses may render the method unfeasible.

Based on the idea of *cone-tracing* introduced in [Ama84] and inspired by various prefiltering based techniques we propose to employ hardware-filtered geometry buffers to reduce the amount of samples (see Figure 4.12(a)). In-



(a) Prefiltered MIP chain

(b) Surface with varying roughness

Figure 4.12: Screen Space Cone Tracing. (a) A geometry buffer is prefiltered using hardware mipmapping. Here only the albedo buffer is shown. (b) By raymarching in screen space along a reflection ray and indexing into the mipmap chain depending on the solid angle, glossy reflections can be efficiently simulated.

stead of casting and integrating multiple rays, a single ray is traced through the scene as usual. At each query into the GBuffer, the length of the ray and its corresponding solid angle (which in turn depends on the surface roughness α) are used to index into a mipmap chain of the GBuffer and query an approximate of the integrated radiance. Error therefore increases with higher glossiness.

A result can be seen in Figure 4.12(b): The tiled floor exhibits different glossy appearances due to different surface roughness for tiles. For moderately rough surfaces this method can be used to augment near-field glossy bounces quickly.

4.4.3 Voxel Cone Tracing

Several of the problems associated with the small volume size of LPVs can be attacked with a fusion of the ideas in the last section. I have presented a superior solution to sample reflections of nearby objects in [Fra14a]. Building on the same general idea of having a volume situated around the virtual object, the resolution is increased to at least 256^3 voxels.

By prefiltering the volume with hardware mipmapping, cones can be traced

in the same fashion as presented in the last section without suffering from the same limitations as screen space methods in general, such as unavailable back-facing geometry etc. The voxelized tree structure decouples geometric complexity from the lighting information and can be traced efficiently on the GPU, hence the name Voxel Cone Tracing (VCT).

Further discussion of VCT details is deferred to Section 5.4.

4.5 Conclusion

In this chapter, I have reviewed and tested several methods to shade virtual augmenting objects according to natural illumination. By classifying and distinguishing AR simulations to support *either* fully dynamic or rigid objects, different combinations of algorithms for each class yield better results. Rigid objects can use precomputed effects that would otherwise introduce performance penalties up to a point where a simulation could not be considered real-time any more.

This chapter has addressed and solved **Problem 1** and **Problem 3** discussed in Section 1.1. The following two contributions were made to do so:

Shading rigid objects If the AR renderer can assume to display a rigid object with static materials under dynamic illumination, surface transfer and materials can be precomputed using the combination of PRT for low-frequency self-shadowing and a Sum of Gaussians to encode materials, presented in Section 4.3. Since the entire shading process essentially amounts to the computation of one light coefficient vector and two simple dot products, this combination is very efficient even on older hardware. While the ability to support measured materials is appealing, PRT offers the ability to precompute diffuse interreflections on the geometry as well. However, these can also be augmented with other methods discussed below.

Shading dynamic objects If the AR renderer has no guarantee that either materials, geometry or both remain static during the simulation, we need to exchange all precomputed transfer with dynamic algorithms. The proposed combination is to use SSAO for visibility, Filtered Importance Sampling for specular reflections and low-frequency irradiance mapping for diffuse transfer, all presented in Section 4.2. Compared to the shading pipeline based on precomputation, two key disadvantages are the overhead for sampling the specular reflectance, which amounts to more texture fetches, and the reduced accuracy when computing visibility, as SSAO does not include any directional components. The latter issue may be addressed by using SSDO [RGS09].

Sampling distant specular and diffuse incident light has been addressed in all solutions with the help of light-probes or fish-eye lens cameras. However, so far virtual objects only reflect their real surrounding, in which other virtual objects are not included. To add these for diffuse inter-reflections, a low-frequency solution such as LPVs can provide bounces off other virtual objects which can be sampled. In the case of glossy and highly specular surfaces, a more sophisticated algorithm is needed to quickly gather samples of light bouncing off other virtual objects. In the next chapter, I will discuss these solutions, which are used not just for shading virtual surfaces correctly, but for relighting real ones as well.

Chapter 5

Relighting Reality

5.1 Introduction

A necessary requirement for a *fusion* of virtual objects with a real context is believable interaction of real and virtual light with real and virtual surfaces. Where the methods of the previous chapter ensure that virtual objects appear

in harmony with their surrounding by transferring real light onto a virtual surface, shadows – which provide the visual cues to locate a virtual object in a real scene – and the mutual indirect interaction of illumination are the necessary details to convince an observer that the rendered result is not merely glued on top of an image but part of the real scene.

In this chapter, I will explore and formulate three novel methods to account for the change in illumination on real surfaces when introducing a new object into a scene. This *relighting* process, with the knowledge of real surface properties, needs to adjust existing illumination by either adding new energy which was reflected off the virtual object or subtracting the surplus still present in the image.

5.1.1 The Relighting Problem

Relighting is the process of reproducing original parameters used to create an image and then recomputing the image under different illumination conditions. This process has many applications: Modern movie-sets for instance often feature entirely virtual scenes which are filmed with actors in a green-screen box. Green surfaces are later replaced with another scene in a compositing step. To create matching conditions on the actors skin and clothing (for instance when she is located in a scene directly beneath a single small light source compared to one with multiple light sources from below), the appearance of the actor is *relit*[1].

Whenever any object is to be shown in *a new light*, its fundamental properties have to be reconstructed to recompute a new image. Since these properties are usually known for augmenting objects, they are merely shaded using the reconstructed lighting conditions of the real scene. At the same time however, they change the illumination conditions for real objects as well. It is therefore necessary to create algorithms which can accommodate for this influence of and between virtual and real objects, relighting real surfaces to match these new conditions. Relighting culminates to an interdisciplinary

[1] The *Lightstage* is a popular example [DHT+00].

fusion between reconstruction algorithms, geometric registration and global illumination.

In Augmented Reality applications, the steps necessary to transform the image for changed conditions in illumination are exacerbated by the real-time constraint: The delivery of a fused image of virtual and real scene elements – with its entire reconstruction and scene reasoning process, as well as a believable illumination computation – has to be executed in very short time window. Several steps in this pipeline are non-trivial problems which, when taken together, amount to a workload which often cannot be reconciled within a window which would be considered real-time.

Contemporary solutions introduce the change in illumination with shadowmapping and real-time global illumination methods by extracting the difference in illumination using Differential Rendering [Deb98]. This design choice has one major flaw: Computation of global illumination must be executed two times; any expenses in the rendering pipeline are doubled by default. Algorithms with high computational costs near the real-time limit are therefore rendered unsuitable for further discussion. This is evident in the scientific publication record: Methods proposed for rasterizers restrict themselves to reproducing diffuse global bounces only, while raytracing based options only handle highly specular cases efficiently. There is currently no solution which can reproduce any type of glossy reflection of virtual elements on real surfaces.

5.1.2 Related Work

Differential Instant Radiosity [KTM+10] makes use of RSMs to transfer indirect light onto surrounding real geometry, and was later extended in [KTWW13] to handle reflected and refracted VPLs, caustics and specular reflective objects. Through differential rendering, first bounce indirect light is extracted and can be added to the background image. To handle a low amount of VPLs without flickering, the authors exploit frame-to-frame coherence and blend illumination from each previous frame by calculating a confidence value for each pixel which depends on the differential of the nor-

mal, illumination and position. In contrast, Lensing and Broll [LB12] use multi-resolution splatting [NW09] to use a high number of VPLs. Two approaches presented in this Chapter do not suffer from inherent flickering and are independent from the RSM sampling rate.

Several clustering methods exist to increase the amount of VPLs without hurting performance. Prutkin et al. [PKD12] importance sample an RSM and combine several VPLs via k-means clustering into area lights. Light Propagation Volumes (LPV) [KD10] inject VPLs generated from an RSM into a small volume and propagate their contribution. Crassin et al. [CNS+11] compress the scene into a high resolution 512^3 sparse voxel octree and use Voxel Cone Tracing (VCT) to gather indirect contribution and cone trace visibility. Due to the high spatial resolution (which is impractical for propagation based schemes) the result does not suffer from large bleeding effects, supports specular indirect bounces and reduces aliasing effects on sharp borders. An observation is that in much higher resolution volumes many blocks in the octree have common configurations and can be compressed with pointer structures to common blocks [KSA13]. The compression however only works for binary voxels and has a high impact on rendering speed.

Kán and Kaufmann [KK12] present a real-time raytracer which is naturally able to render highly specular reflections and refractions of virtual objects. A study confirms the importance of such effects on how users perceive visual coherence and quality of the augmented image. In a follow-up publication [KK13] the authors present an extension to their method to support diffuse indirect reflections with a modification of Irradiance Caching. This Chapter presents methods which support first bounce reflections with surface BRDFs of varying roughness - mirror-like, glossy and diffuse - at lower evaluation cost in a unified and scalable relighting framework. Because my methods rely on volumetric simplification of the real scene, a tradeoff between visual fidelity and evaluation speed is possible by tuning the size of the volume depending on available GPU memory.

Recently, Rohmer et al. [RBDG14] have demonstrated interactive global light transport for AR relighting on mobile devices. The computation of the final image is shared between a server and the mobile device: Captured illumina-

tion is projected onto reconstructed geometry, split into high-frequency area lights and low-frequency PRT projection and transferred to a mobile client which recombines both and extracts the relighting information with Differential Rendering (from now on referred to as Differential Radiance Atlas (D-RA) method). The virtual object is assumed to be rigid.

Grosch et al. [GEM07] compute scaling factors for subdivided distant illumination regions which are used to linearly combine multiple basis Irradiance Volumes. The final volume is used to query illumination for a surface point of a virtual object and can transfer indirect illumination from the virtual scene. This however presumes that the entire real reconstructed scene is rigid.

Many approaches employ regular shadow-mapping techniques to cast virtual shadows on real surfaces, for instance [HRKP04, FJ08b, KTM+10, LB12]. Similar to regular shadow mapping, key light sources are reconstructed or placed manually and a suitable shadow mapping technique such as PCSS or VSM is used to render a shadowed region onto a reconstructed surface. Some methods such as [HRKP04] simply alpha blend between pixels of the background image and a binary shadow projection, which corresponds to an incorrect darkening of the area. This method will fail to produce a realistic image when multiple light sources with different wavelengths are present.

In a similar manner to shadow mapping, Haller et al. [HDH03] and Hughes et al. [HRKP04] have demonstrated the use of shadow volumes in AR.

Nowrouzezahrai et al. [NGM+11] factorize a low-order spherical harmonic representation of a light-probe into a directional and a global component. The directional component can be used as regular point light while the global term is applied using matrix radiance transfer. Shadows are cast using PCF shadow mapping.

Gibson et al. [GCHH03] compute multiple paths between patches and blend hard-edge shadow maps to correctly smooth out non-contact shadow borders.

In a pre-process Kakuta et al. [KOI05] discretize the hemisphere around a virtual object into a polyhedron. A set of basis shadow maps are computed from the position of each vertex of the polyhedron, which are linearly combined with coefficients computed from the luminance per unit area of a hemispherical image of each face. The extended and very similar Radiance Transfer Fields [PWL+07] have been proposed to be used to transfer indirect light to surrounding real geometry. However, both methods requires the virtual object to be rigid.

5.1.3 Contribution

In this chapter, I will formulate three novel methods to account for the change in illumination on real surfaces when introducing a new object into a scene. These methods are:

- AR Object Occlusion Fields for rigid objects [FKOJ11]
- Delta Light Propagation Volumes [Fra13a]
- Delta Voxel Cone Tracing [Fra14a]

Whilst relighting algorithms are specifically aimed at computing light for anything *but* the virtual object, the last two proposals are derived from regular GI shading algorithms which can be used to add local, *near-field* light bounces from either reconstructed real objects or other virtual ones onto the augmenting objects surface.

At the end of the chapter, a final discussion will compare all three methods to one another and the current state of the art in relighting of the scientific literature. Since the aim is to create a relighting solution that is both realistic and efficient, these two objectives will guide the judgment of whether or not any one of these proposed methods is a good candidate for relighting and better suited than current competing algorithms. The evaluation will look at efficiency, the range of supported types of interreflections and other strengths and weaknesses.

5.2 AR Object Occlusion Fields

Rigid objects, under static illumination, always block off light from the same directions and therefore always cast identical shadows. Under low-frequency lighting conditions, shadows only change smoothly with varying lighting conditions. Several methods exploit this behavior by precomputing shadows from multiple angles around a rigid object and reconstruct the shadow for any lighting condition from the collected samples. A good basis function can reduce the memory footprint of the retained basis shadow samples. Furthermore, re-projecting the reconstructed shadow onto another arbitrary surface from its initial sample needs to be efficient.

To simulate virtual shadowing on real geometry caused by natural illumination we propose a modification of Object Occlusion Fields (OOF) in [FKOJ11] called AR-OOF.

5.2.1 Algorithm Overview

A basic requirement of PRT is that objects remain rigid so that their transfer functions on the surface can be precomputed. OOFs [ZHL+05] relax the rigidness requirements *between* objects so that they can behave dynamically under affine transformations and therefore might affect other objects with precomputed transfer (for instance by blocking off light). Shells around the object O with sparse sample positions $\mathbf{q}$ save the binary occlusion information introduced by the presence of the object as spherical harmonic vectors $t_{\mathbf{q},i}$, as displayed in Figure 5.1. This influence is a spherical representation of L_Δ, with the exception of carrying only antiradiance. Since each shell is essentially a sphere around the object, we upload these sampled coefficient vectors $t_{\mathbf{q},i}$ as low resolution latitude-longitude textures to the GPU. Access on the GPU to this texture automatically handles interpolation for unsampled spaces on the AR-OOF shell.

When a shell collides with another surface, at each intersection between the shell and the surface their respective transfer vectors are combined. The

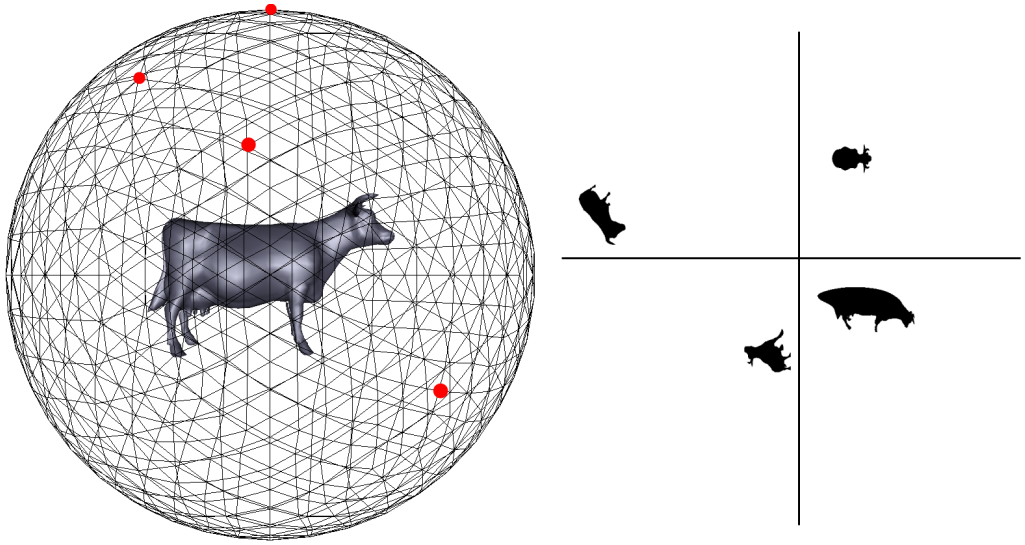


Figure 5.1: Left: An AR-OOF shell is sampled around the object, e.g. at the highlighted vertices on the sphere geometry. Right: For each sample a binary occlusion function is saved and compressed by projection into spherical harmonics.

transfer from the shell adds the occlusion by the object to the transfer of the surface. This combination is achieved with the help of a triple product integral.

5.2.2 Triple Products

In this section I will shortly introduce the term *triple products*. Given three functions $O(x)$, $T(x)$ and $E(x) = O(x)T(x)$ on a domain Ω, coefficients for any given orthonormal basis Φ can be calculated with:

$$O_j = \int_\Omega O(x)\Phi_j(x)dx \tag{5.1}$$

$$T_k = \int_\Omega T(x)\Phi_k(x)dx \tag{5.2}$$

Coefficients for $E(x)$ are derived in the manner of Equation (5.3), where i, j and k are indices of the basis components.

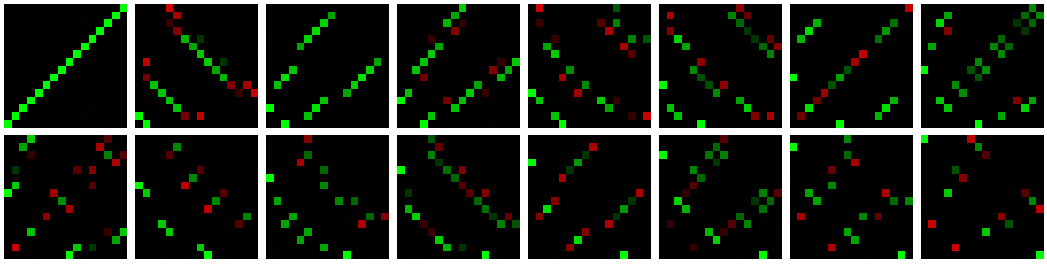


Figure 5.2: Precomputed SH tripling coefficients. Each texture k stores $i \times j$ coefficients.

$$
\begin{aligned}
E_i &= \int_\Omega \Phi_i(x) E(x) dx = \int_\Omega \Phi_i(x) \left(O(x)T(x)\right) dx && (5.3)\\
&= \int_\Omega \Phi_i(x) \left(\sum_j O_j \Phi_j(x)\right) \left(\sum_k T_k \Phi_k(x)\right) dx && (5.4)\\
&= \sum_j \sum_k O_j T_k \int_\Omega \Phi_i(x) \Phi_j(x) \Phi_k(x) dx && (5.5)\\
&= \sum_j \sum_k O_j T_k C_{ijk} && (5.6)\\
&= \sum_j O_j T_{ij} && (5.7)
\end{aligned}
$$

The term C_{ijk} denotes a *triple product integral* or *tripling coefficient*. For spherical harmonics these values correspond to the well studied Clebsch-Gordan coefficients[2], but there is no general formula to analytically derive these coefficients for any basis Φ. Haar Wavelet tripling coefficients are presented in [NRH04]. Depending on the method of calculation, surface transfer for any point p can be saved as either a transfer vector $\dot{t}$ or a transfer matrix $\mathbf{T}$, as shown in Equation (5.7).

5.2.3 Implementation

At runtime, after each transformation, each PRT object (i.e., each surface with precomputed transfer vectors) is tested for collision with an AR-OOF

[2] A straightforward practice is to express these with the help of Wigner 3j symbols. An implementation is available in [Gou09].

shell of another object. This can be realized through a simple radius collision test for each shell. Colliding geometries will adjust their surface transfer $t_{\mathbf{p},i}$ at surface point $\mathbf{p}$ by combining it with the *delta coefficient vector* $\widehat{t_{\mathbf{q},i}}$ (i.e., $t_{\mathbf{q},i}$ in the global frame) carrying the occlusion information on shell position $\mathbf{q}$ via triple product on the GPU. To do this efficiently, we precompute the sparse tripling coefficient matrix and store it in a 3D texture (see Figure 5.2).

The local transfer is now modified by the change introduced through the colliding AR-OOF. The augmented transfer vector $t^a_{\mathbf{p},k}$ can be combined in a double product with the light vector that is rotated with $\mathbf{R}_\mathbf{p}^{-1}$ into the local tangent space. Equation (5.9) presents the final formula to merge diffuse transfer with a colliding AR-OOF transfer vector and relight the scene with light coefficient vector $\dot{l}$ accordingly.

$$t^a_{\mathbf{p},k} = \sum_j \sum_i t_{\mathbf{p},i} \widehat{t_{\mathbf{q},j}} T_{ij} \tag{5.8}$$

$$L(\mathbf{p}, \vec{\omega_o}) = \left\langle \left(\mathbf{R}_\mathbf{p}^{-1} \cdot \dot{l} \right), \dot{t^a_\mathbf{p}} \right\rangle \tag{5.9}$$

Applying this solution to a MR simulation with access to a light-probe or a (fish-eye) light camera and a depth camera does not work at first. The incident light captured by the light-probe or (fish-eye) light camera is transformed into a coefficient vector. A simple dot product of a surface transfer vector from a virtual object with the light vector yields the correct shading under real incident light, however the surface onto which the virtual object will cast its shadow is unknown and therefore transfer coefficients cannot be determined. Instead, we assume real surfaces to have purely local diffuse BRDF f_γ and precalculate basic diffuse transfer coefficient vectors for regular sample normal directions $\vec{n}$ and save them in a cache:

$$f_\gamma = \frac{\rho_d}{\pi} \tag{5.10}$$

$$t_i = \int_\Omega f_\gamma \langle \vec{n}, \vec{\omega} \rangle_+ \Phi_i(\vec{\omega}) d\vec{\omega} \tag{5.11}$$

For each surface point, its proper set of coefficients can be queried from the cache using the surface normal as an index.

At runtime, we derive world space positions $\mathbf{p}$ of each pixel in the real image with the help of the depth map provided by a depth-sensor (such as a Microsoft Kinect) and calculate a normal vector $\vec{n_\mathbf{p}}$ for it based on its neighbors. If the position $\mathbf{p}$ is inside a shell, we interpolate an occlusion vector $\widehat{t_{\mathbf{q},i}}$ from the OOF and select a transfer vector of the real surface point $t_{\mathbf{p},i}$ from the cache we have built up earlier by indexing into it with $\vec{n_\mathbf{p}}$ and apply Equation (5.6) to calculate the combined transfer vector. The double product of the light vector and the augmented transfer vector yields a shadow attenuation value that can be multiplied with the fragment's color.

5.2.4 Discussion

Given a rigid object, AR Object Occlusion Fields provide high flexibility in an AR simulation when casting shadows.

Much like PRT based on the SH basis, AR-OOFs can only provide low-frequency transfer at reasonable memory costs. In Figure 5.3 a Cow object is shown next to two different surfaces. In both cases, the geometric distortion from the new parametrization onto which the shadow is projected has almost no visible artifacts due to the very smooth outline of the shadow. However, this is at the same time a downside of this method, as it does not preserve any geometric details.

AR-OOFs can be trivially extended to additionally provide low frequency indirect light bounces of virtual geometries onto real surfaces with the help

(a) (b)

Figure 5.3: Results. The cow casts a very low-frequency shadow once it is near another surface. Although the method works with natural illumination, almost all shadow-features of the object are lost.

	# Vertices			
	100	1000	100000	307200
Time	3.22	4.45	27.39	43.20

Table 5.1: Timings for different numbers of triple product operations. The last column represents the amount of computations necessary on a default 480p Kinect depth image producing as many reconstructed real surface vertices as there are pixels.

of Radiance Transfer Fields [PWXLPB07]. This enhancement will result in a full representation of L_Δ.

Table 5.1 lists the time for a simulated plane surface which picks a set of surface transfer coefficients based on each surface normal, then computes Equation (5.9) for a surface point with the incident light from the irradiance coefficients. Because this computation takes place on a per-vertex level, the table lists timings taken for surfaces with different numbers of vertices. The last column represents a surface created from every pixel of a depth map at 480p. The computation of the triple product at this resolution is very costly and crosses the threshold for real-time behavior. AR-OOFs are therefore only suitable for pre-reconstructed geometries with compact representations (i.e., less than 100k vertices).

5.3 Delta Light Propagation Volumes

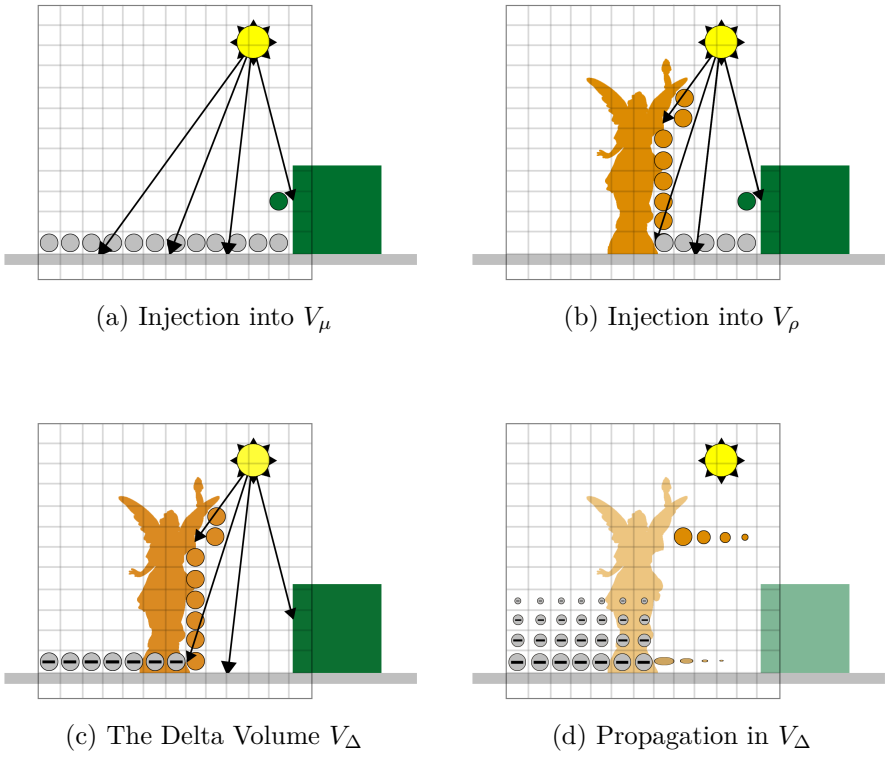
AR Object Occlusion Fields and extensions of it can be used to correct real scenes for augmenting objects, but their basic requirement is that the object remains rotationally invariant. Furthermore, because of the choice of basis to encode light transfer, the cached diffuse coefficients which are picked as representatives of the real surface transfer are predetermined and do not allow consideration of varying self-shadowing and other behavior on the real surface.

I attack these limitations with a novel AR global illumination method in [Fra13a]. The challenge is to create an algorithm which can transfer shadows and indirect bounces from an augmenting object onto real surfaces whilst considering arbitrary changes of either real or virtual geometries in real-time, i.e., the costly re-evaluation of AR-OOFs for dynamic geometries and the low spatial resolution should be solved by this algorithm.

5.3.1 Algorithm Overview

Inspired by Radiance Transfer Fields [PWL+07] and Differential Rendering, I model a Delta Radiance field L_Δ by extending Light Propagation Volumes [KD10]. This new representation will be used for Augmented Reality light transfer. A LPV $V^n(\mathbf{j}, \vec{\omega_o})$ is a volume of n^3 voxels $\mathbf{j}$ which store approximate propagated diffuse indirect intensity which can be queried for a direction $\vec{\omega_o}$. Similarly to Irradiance Volumes, a LPV covers part of a 3D scene and can be queried for each position in space to retrieve the directional indirect light contribution at this position.

The evaluation of an LPV is done entirely on the GPU: A RSM is computed for a light source, and VPLs generated from it are *injected* into their respective voxels inside the volume (if their position is covered by the volume). The directional distribution intensity of a VPL is saved by converting it into low-frequency, second order spherical harmonic (SH) coefficients. An SH value of



(a) Injection into V_μ (b) Injection into V_ρ

(c) The Delta Volume V_Δ (d) Propagation in V_Δ

Figure 5.4: Delta Light Propagation Volume construction. (a) A regular LPV V_μ without the virtual object after injection with R_μ and similarly (b) LPV V_ρ injected with R_ρ. (c) The residual delta between of $V_\rho - V_\mu$. Indirect light from the virtual object remains as positive contribution, while the shadowed space behind it now has negative values. (d) The propagated delta $R_\rho - R_\mu$ yields the DLPV.

band b has b^2 coefficients, therefore one can save the entire volume in three volumetric RGBA textures, one for each color channel.

To illustrate this method we can now think of two independent LPVs: V_ρ created from R_ρ and V_μ created from R_μ. Applying Equation (3.11), the difference between both volumes, the *Delta Light Propagation Volume* (DLPV) $V_\Delta = V_\rho - V_\mu$, contains only the change in illumination caused by the introduction of the virtual object O into the scene. In Figure 5.4(c) we can observe that this differential not only includes indirect change of illumination but also antiradiance (negative voxel values) which implicitly creates shadowed areas when superimposed onto another radiance field.

A naive approach is to create both volumes on the GPU, query both for their radiance contribution for each point in space and add the difference to a background image or radiance field. Compared to regular rendering, this approach doubles the memory footprint and requires twice the computation time to extract the illumination difference. Instead, I choose to create one volume which will *propagate the change in illumination* directly, hence the name *delta-volume*. To this end, I alter the injection phase of a regular LPV and calculate the differential at the level of VPLs directly, avoiding the the memory and propagation cost for one of the volumes.

I start by reconstructing one or more real direct light sources with position and orientation (e.g., tracked or extracted from a hemispherical image), as well as the real geometry of the scene with the help of a depth sensor or a pre-reconstructed model. I use the reconstructed light sources to furthermore evaluate diffuse surface parameters of the real scene. Two RSMs are created per light source: One for the real scene only, and one together with the augmented virtual object. The differential VPLs of both RSMs of each light source are injected into a *delta-volume* V_Δ (see Figure 5.4(c)).

The construction of a DLPV, laid out in Figure 5.4 is divided into the following steps:

Split-inject For each light source I render the virtual object and the reconstructed real geometry of the scene into an RSM R_ρ. The same process is repeated for a second RSM R_μ which contains only the reconstructed real geometry. Each pixel in both RSMs is injected: One from R_ρ the regular way, and one from R_μ at the corresponding position negatively. This yields the basis volume of a DLPV.

Propagation After injection, the DLPV construction proceeds by propagating the injected radiance to its neighboring voxels. This operation can be reformulated from a scatter to a gather process, where each voxel collects neighboring radiance. To query the correct value, the solid angle over the shared voxel side is evaluated from the spherical harmonic coefficients of

one voxel and then transformed back and added on top of the current voxel coefficients.

Direct Inject By injecting not only indirect, but also direct light, I model both operators $\mathbf{T}_{\Delta_1}$ and $\mathbf{T}_{\Delta_2}$ with one representation.

Query In the last step during image composition, the DLPV is simply queried for a corrective factor on each point on the surface. The current position is transformed into DLPV space and, using the inverse of the current surface normal, the coefficients at this position are evaluated for incident radiance. The result from V_Δ is simply added to the existing background image at its intersection with the real geometry. A sample can bee seen in Figure 5.5.

5.3.2 Construction

Indirect Change $\mathbf{T}_{\Delta_2}$ The intensity distribution $I(p, \vec{\omega_o})$ of a VPL created from pixel p of an RSM with normal $\vec{n}_p$, reflected flux Φ_p and direction $\vec{\omega_o}$ that is initially injected into a voxel of an LPV is:

$$I(p, \vec{\omega_o}) = \Phi_p \langle \vec{n_p}, \vec{\omega_o} \rangle_+ \tag{5.12}$$

For a DLPV, I instead inject:

$$I_\Delta(p, \vec{\omega_o}) = I_\rho(p, \vec{\omega_o}) - I_\mu(p, \vec{\omega_o}) \tag{5.13}$$

The outcome of this operation is that VPLs visible in both RSMs are eliminated. New VPLs created from bouncing off the surface of the virtual object O are added, while VPLs now blocked by it appear as negative contribution.

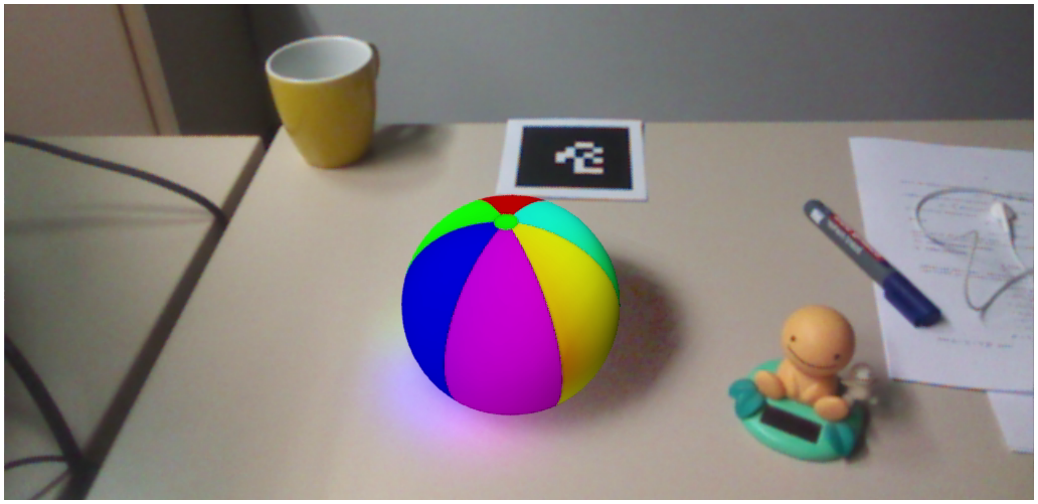


Figure 5.5: Using a DLPV to display shadows cast by blocking direct light as well as indirect bounces. The above image exaggerates the indirect value for better illustration.

To avoid self-illumination from VPLs injected into voxels which also contain the surface they originate from, I shift the position of the VPL inside the volume by half the size of a voxel along its normal before determining the voxel into which the value will be injected, as proposed in [KD10]. The same procedure is safe-guarding against discretization errors, where VPLs appear behind the surface they are supposed to bounce off from.

Direct Change $\mathbf{T}_{\Delta_1}$ After injection and propagation of indirect change, the volume contains indirect bounces of light with subtle shadows from blocked indirect sources. However, shadows are usually cast by blocking direct incident light, which is why I inject direct light into the volume as well.

For each reconstructed direct light source L_γ I use the same RSMs R_ρ and R_μ to determine surfaces that are directly lit. Analogously to VPLs, for each pixel in the RSM R_ρ I reconstruct the surface position but instead use the negative incident direction $\vec{\omega_i}$ and flux Φ_r of a reconstructed real light source L_γ (from which the RSM was created) to compute the SH value. This value is injected just like a VPL created from that pixel, but with a safe-guard distance of half a cell in direction $\vec{\omega_i}$ of L_γ.

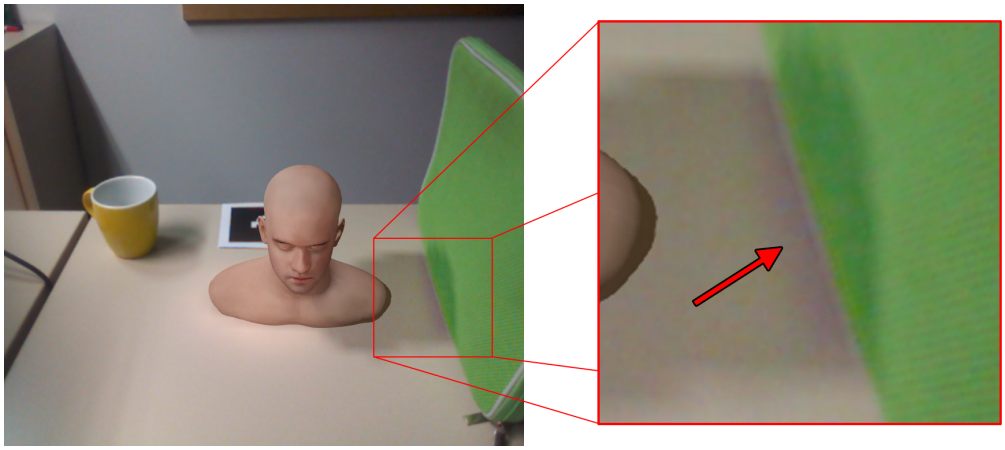


Figure 5.6: The low resolution and frequency encoding of DLPVs cause bleeding artifacts. Here the negative indirect green bounce that needs to be subtracted from the now shadowed real surface below is overestimated, leading to a purple mismatch.

5.3.3 Reducing Shadowing Artifacts

Voxels $V_\Delta(\mathbf{p})$ which have a value $I_\Delta(p, \vec{\omega}) < 0$ in direction $\vec{\omega}$ after injection can cause severe shadow bleeding artifacts when propagated. Because the initial injection for indirect bounces is associated to the reconstructed material where the light hits the surface, propagating these values can affect the surrounding real geometry and subtract light incorrectly due to the low spatial approximation of DLPVs. For instance, in Figure 5.6 one can see thin purple line along the edge of the green briefcase, which is the result of negative green contribution onto the white desk.

Shadow bleeding artifacts cannot be avoided entirely and are visible mostly along edges of real reconstructed geometry or bleeding out from below virtual objects. Since this error is related to the light bleeding artifact in [KD10], I use directional derivatives of the intensity distribution to dampen these values. The same dampening factor also helps to contain self-shadowing.

5.3.4 Merging DLPVs with the Real Scene

When compositing the real scene with the rendered virtual result, a G-Buffer $G(p)$ is used which contains the reconstructed albedo of the real scene as well as the albedo of the virtual object, a binary mask to distinguish real and virtual elements, and normals for each pixel p. First the position in real space $\mathbf{p}$ is reconstructed for each pixel, as well as its normal $\vec{n}_{\mathbf{p}}$. Furthermore, a real background image buffer can be accessed with $B(p)$. If a LPV model of real light bounces $\left(\mathbf{T}_{\mu}^{2} + \mathbf{T}_{\mu}\right) L_e \approx B(p)$ is accurate then $\left(\mathbf{T}_{\Delta_2} + \mathbf{T}_{\Delta_1}\right) L_e$ will likewise accurately adapt $B(p)$ for the necessary changes in illumination when introducing a new object.

The reconstructed diffuse surface properties are identified by a BRDF f_γ. For n real, reconstructed light sources L_γ, the virtual object is rendered the regular way by adding direct and indirect contribution of V_ρ to include bounces from virtual and real geometry on the object's surface (see Figure 5.8). Transfer operators $\mathbf{T}^n$ for pixels p identified as virtual are defined as follows:

$$\mathbf{T} L_e = \sum_{i}^{n} f_r\left(\mathbf{p}, \vec{\omega_i}, \vec{\omega_o}\right) L_\gamma\left(\mathbf{p}, -\vec{\omega_i}\right) \left\langle \vec{n_{\mathbf{p}}}, \vec{\omega_i} \right\rangle_{+} \tag{5.14}$$

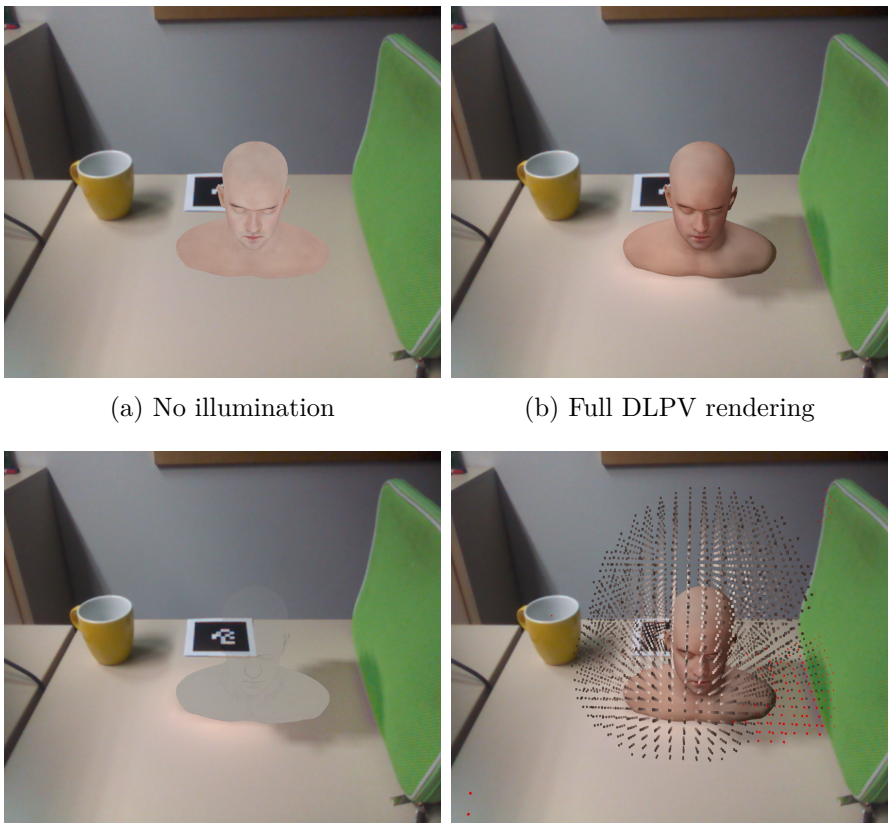
$$\mathbf{T}^2 L_e = f_r\left(\mathbf{p}, \vec{\omega_i}, \vec{\omega_o}\right) V_\rho\left(\mathbf{p}, \vec{n}_{\mathbf{p}}\right) \tag{5.15}$$

For all real pixels p of the background image, they instead query the DLPV and add its value onto it. The delta transfer operator $\mathbf{T}_{\Delta_1}$ and $\mathbf{T}_{\Delta_2}$ are combined in the volume as follows:

$$\left(\mathbf{T}_{\Delta_1} + \mathbf{T}_{\Delta_2}\right) L_e = V_\Delta\left(\mathbf{p}, \vec{n}_{\mathbf{p}}\right) f_\gamma\left(\mathbf{p}, \vec{\omega_i}, \vec{\omega_o}\right) \tag{5.16}$$

5.3.5 Implementation

I have implemented the system described in Section 5.3.1 using Direct3D 11. A Microsoft Kinect camera is used to capture the background image, and a UEye UI-2230-C camera in conjunction with a fish-eye lens to capture surrounding real light. A marker based tracking approach through OpenCV [Bra00] is used to geometrically register a virtual object. For my tests, when the camera was very close to the scene the Kinect depth buffer could not be used to reconstruct it. In these instances I therefore registered a manually reconstructed model of the real scene.



(a) No illumination (b) Full DLPV rendering

(c) DLPV effects only (d) Volume visualization

Figure 5.7: Infinite Head model inserted into a real scene with one reconstructed light source. (a) A virtual object is inserted without illumination. (b) Visible first bounce around the base as well as low resolution shadow. (c) Indirect effects without virtual object for better visualization. (d) Visualization of the DLPV (red dots indicate negative values).

Figure 5.8: A textured bust of Joseph von Fraunhofer (left) is inserted into the scene, receiving red indirect light from the ground and casting a slight shadow.

First the reconstructed geometry is rendered with the virtual object into an RSM R_ρ of size 512^2 pixels which are all injected as VPLs into V_Δ^{32}. The process is repeated only for the real scene geometry to create an RSM R_μ, which is injected as negative intensity using subtractive hardware blending. To avoid slight flickering artifacts from inconsistent injections when rotating the object, the volume orientation is kept synchronous. 32 propagation steps are used to distribute the light inside the DLPV before accessing it in a deferred rendering step to assemble an image. A postprocessing pipeline can optionally add SSAO or bloom effects to bright spots and will tonemap HDR values and gamma correct the output. An example can be seen in Figure 5.7.

Figure 5.8 shows a comparison of two busts: A real stone bust on the right, and the bust of Joseph von Fraunhofer on the left. The virtual object on the left casts a low-frequency shadow on the ground while at the same time receiving a light red tint from light reflected of the red paper ground.

Rendering Step	Infinite Head	Cornell Box	Stanford Lucy	Stanford Armadillo
R_μ	0.123	0.0629	0.114	0.0675
R_ρ	0.152	0.0626	1.174	0.816
V_ρ Inject	0.367	0.3674	0.4728	0.368
V_Δ Indirect Inject	0.734	0.7345	0.7808	0.734
V_Δ Direct Inject	0.73	0.73	0.837	0.73
V_ρ Propagation	3.619	3.664	4.055	3.652
V_Δ Propagation	3.833	3.777	4.437	3.913
Phantom Scene	0.3	0.26	0.31	0.32
Virtual Object	0.1	0.0198	1.14	0.894
Deferred	1.539	1.526	1.972	1.551
$\sum$	11.497	11.2042	15.2926	13.0455
Generating V_Δ	5.572	5.367	7.3428	6.2605

Table 5.2: Detailed timings per frame in milliseconds taken for each step of the pipeline, with 512^2 VPL injections at 32 propagations. The highlighted rows display the effective time to generate the DLPV V_Δ. In the last row the sum of these operations add up to roughly 6 ms on average.

I have gathered timings for the entire pipeline on a test system with an Intel i7 X980 and a NVIDIA GTX 470 in Table 5.2. After injection, the required propagation time does not differ from a regular LPV. For each reconstructed light source two RSMs need to be calculated. While LPV and DLPV propagation stay approximately the same for each model, on average the delta injection consumes twice the time of a normal injection.

Increasing the size of both RSMs can lead to better sampling of VPLs. The time consumed for injecting every pixel of a RSM into a regular volume as well as a DLPV is listed in Table 5.3. Up to 1024^2 sampled VPLs amount to a reasonable investment for real-time purposes. Since the injection step essentially clusters VPL contribution into a fixed amount of voxels, the following propagation remains unaffected.

Better spatial sampling of the radiance field can be achieved by increas-

RSM Size	$\mathbf{V}_\rho$	$\mathbf{V}_\Delta$ Indirect	$\mathbf{V}_\Delta$ Direct
256^2	0.091	0.184	0.18
512^2	0.462	0.862	0.895
1024^2	2.108	4.667	4.494
2048^2	10.74	21.35	20.215

Table 5.3: Timings taken for a full VPL injection (i.e., every pixel is used as a VPL) when varying the size of both RSM R_μ and R_ρ. The test scene contains the Stanford Lucy model and a planar ground. All values are in milliseconds.

		Propagation Steps				
		16	32	64	128	256
$\mathbf{V}_\Delta$ **Size**	16^3	0.712	1.815	-	-	-
	32^3	2.342	4.552	8.857	-	-
	64^3	11.148	22.021	43.23	85.089	-
	128^3	79.96	158.36	304.57	625.89	1250.5

Table 5.4: Timings taken for varying volume sizes of $\mathbf{V}_\Delta$ with varying numbers of propagation steps and the cost in milliseconds.

ing the resolution of the DLPV. The impact on performance, listed in Table 5.4, furthermore depends on the number of propagation steps used to distribute light inside the volume: A higher resolution DLPV implicitly requires more propagation steps for the same distribution coverage. At a size of 64^3 and 32 steps the cost of the propagation dominates the entire rendering pipeline.

I now compare my method to the multi-resolution splatting method by Lensing and Broll [LB12]. Naive sampling of VPLs in Instant Radiosity can force high VPL counts to avoid artifacts, which in turn have a high impact on rendering speed. The multi-resolution splatting method, based on a frequency analysis of the current view space, effectively importance samples regions of higher interest and therefore reduces the amount of VPLs needed to render an artifact-free image.

Timings for the multi-resolution splatting rendering pipeline, displayed in

VPLs	$\mathbf{d}_{normal}$ 10°	5°	1°
1024	5.81	6.32	6.45
4096	9.8	11.9	11.9
16384	11.23	13.5	13.69

Table 5.5: Multi-resolution splatting method timings taken with a fixed d_{depth} of 1 cm for varying VPL counts and d_{normal} degrees in milliseconds.

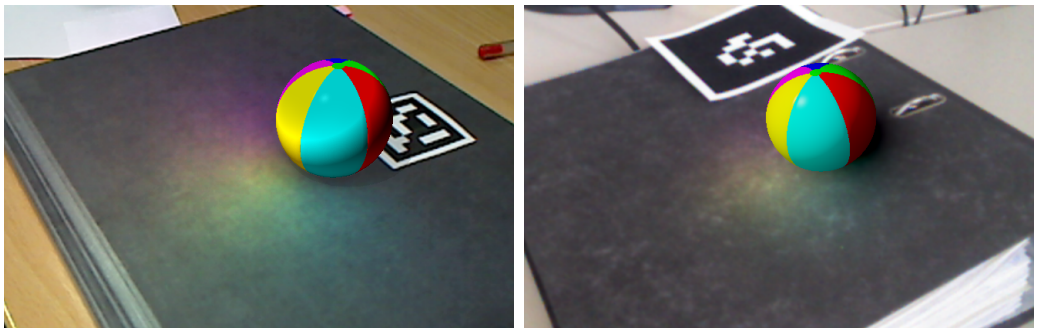


Figure 5.9: Visual comparison between multi-resolution splatting (left) and DLPV rendering (right): The multi-resolution splatting image was rendered with 4096 VPLs, a d_{normal} of 10° and d_{depth} of 1 cm in 9.8 ms. The DLPV image was produced with 32 propagations and 512^2 in 9.6 ms.

Table 5.5, were measured on a NVIDIA GTX 285, a comparable match to the NVIDIA GTX 470 used to measure the DLPV time.

In Figure 5.9 I set up a small test scene with a beach ball. The beach ball features differently colored slices and can lead to temporal inconsistencies when animated or when the viewer camera is moving. In this case the VPL distribution has to be recalculated and can lead to slight flickering. Lensing and Broll report that at a VPL count of 4000 or higher is needed for the beach ball scene to suppress flickering in the animation. More complex scenes require higher VPL counts. In contrast, by clustering 512^2 VPLs into a volume in $\sim$ 2.3 out of 9.6 ms to render the image, a DLPV with 32 propagations handles a VPL count two orders of magnitude higher at the same rendering speed. DLPVs are therefore not constrained by the number of injected VPLs. One should note however that by using this clustering method, many VPLs can get blurred together and may loose details still visible in the

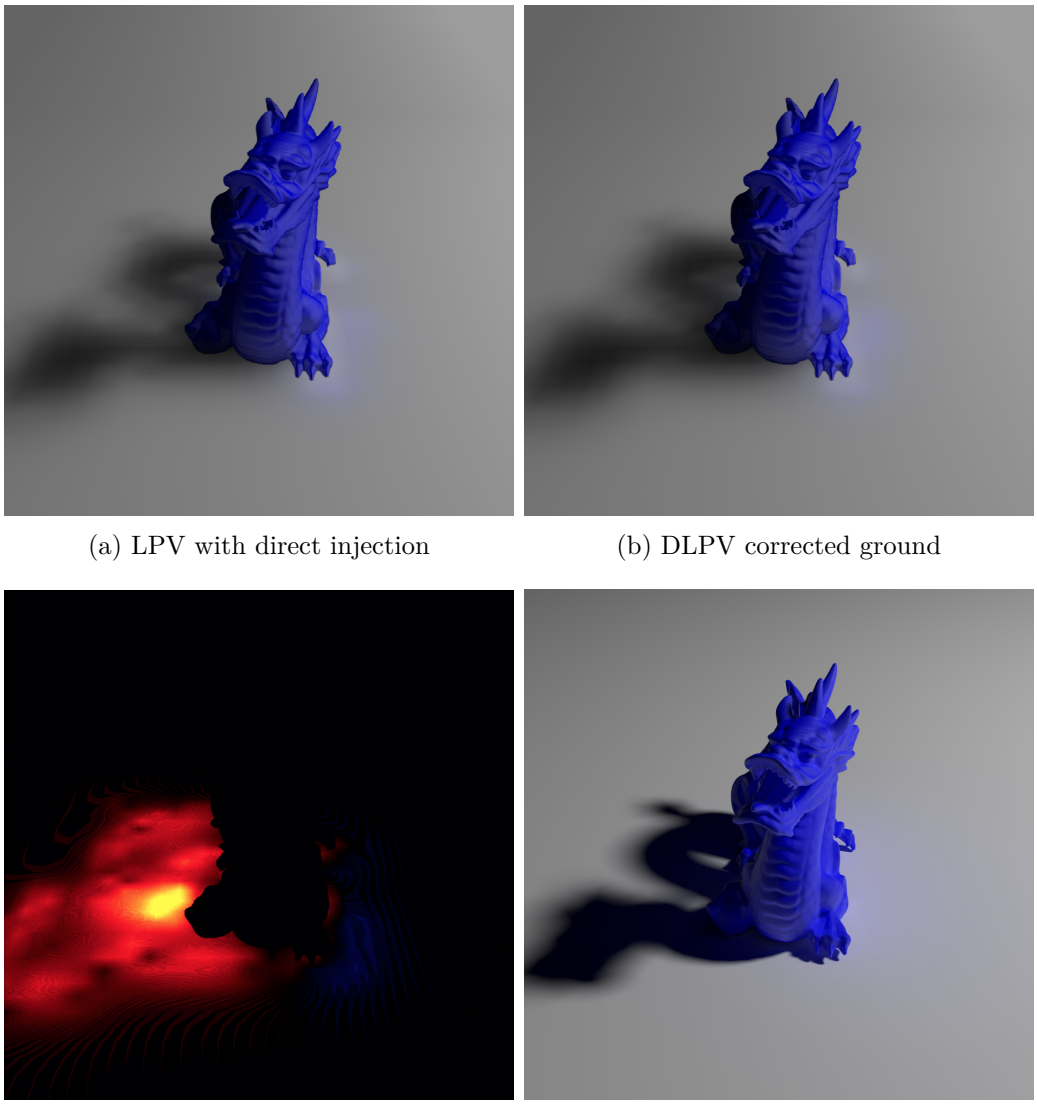
multi-resolution splatting method. Another difference between both methods is that the propagation distance in DLPVs is limited to the number of steps and the size of the volume, whereas regular VPL accumulation in Instant Radiosity is not limited by distance factors other than the physical falloff.

5.3.6 Discussion

When rendering V_ρ to add indirect bounces from real geometries to the virtual object, one could argue to simply evaluate the difference of V_ρ and V_μ in a shader instead of calculating a DLPV. DLPVs however contain less self-illumination errors than a differential of two LPVs, since VPLs which do not contribute to the change of illumination are eliminated during injection. Decoupling both volumes has the additional benefit that they can be rendered at different resolutions without introducing artifacts.

DLPVs can be used in conjunction with other shading methods used to calculate the surface transfer on the virtual object O presented in the last chapter. For instance Precomputed Radiance Transfer [SKS02] can be used to relight rigid objects on the fly. In this case the delta injection saves the bandwidth and propagation cost of an extra LPV V_ρ.

In Figure 5.10(a) a virtual test scene has been rendered with an LPV which also contains direct light propagated for five further steps. In Figure 5.10(b) the ground plane has been rendered into an LPV and has been superimposed with a DLPV created for the dragon. The enhanced squared difference 5.10(c) between both shows the error introduced by using DLPVs: Apart from errors which are caused by aliasing (i.e., propagation of values into wrong voxels because of low DLPV resolution), self-illumination issues from neighbor voxels bleeding into shadowed areas or indirectly lit areas cause slightly more energy to be present in the LPV rendering. These neighboring values are eliminated during the injection phase of the DLPV and are therefore not corrected.



(a) LPV with direct injection

(b) DLPV corrected ground

(c) Squared error (8x)

(d) Ground truth (PBRT)

Figure 5.10: An error estimation for DLPV rendering. (a) 64^3 voxel LPV rendering with additional direct light injected and propagated for five steps. (b) An LPV of the ground corrected with a DLPV. (c) The squared error enhanced by factor eight between image (a) and (b) (negative errors in yellow-red, positive in blue-white, red and white pixels equal higher error). (d) Ground truth path traced with PBRT.

5.4 Delta Voxel Cone Tracing

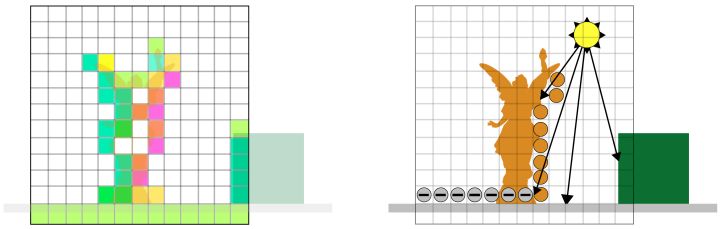
Delta Light Propagation Volumes provide highly temporally coherent GI relighting solution at very low cost. They can be used to add diffuse bounces to a scene, irrespective of it being static or fully dynamic. However, two choices in the structure of DLPVs leave open the same low-frequency encoding limitation as already seen in AR-OOFs in Section 5.2. Due to the propagation-scheme, DLPVs only perform reasonably well at low resolutions. It is also unclear how exactly the number of propagation steps relate to the physical falloff of radiance. Combined with the spherical harmonic encoding of only four coefficients, the propagated light blurs out geometric details and suffers from bleeding artifacts through thin objects, back-scattering and an overall estimation bias.

To remove these limitations, I present a novel global illumination relighting algorithm based on Cone Tracing in [Fra14a]. The idea is to exchange the propagation scheme with cone tracing on voxels, which allows encoding the scene in much higher resolution volumes. This in turn can be used to remove large bleeding artifacts and additionally support glossy and specular reflections of virtual objects on real surfaces.

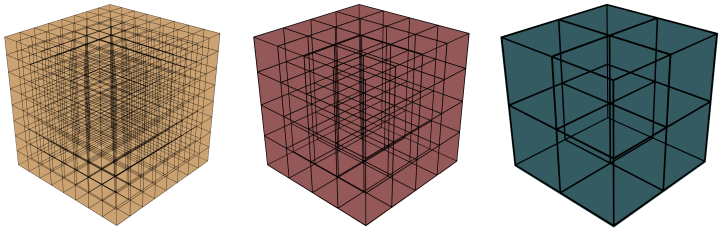
5.4.1 Algorithm Overview

I model a Delta Radiance Field L_Δ with a high resolution, prefiltered hierarchical volume $V_\mu^n(j)$ of n^3 voxels j which store first bounce radiant intensity. VPLs generated from an RSM are *injected* at their respective position inside the volume. The directional component, i.e., the surface normal at the first bounce, is stored in a secondary volume $V_\eta^n(j)$ of the same size.

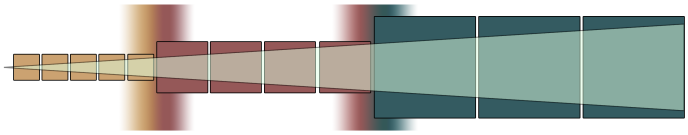
Analogously to Section 5.3.1, two volumes V_ρ and V_μ are injected with VPLs from two different RSMs: V_μ from an RSM R_μ of a scene and V_ρ of another RSM R_ρ of the same scene with an additional object O, from which I create one *delta-volume* $V_\Delta = V_\rho - V_\mu$.



(a) Voxelization into V_η (b) Split-inject into V_Δ



(c) Filtering of $\mathbf{V}$



(d) Cone-tracing on $\mathbf{V}$

Figure 5.11: DVCT overview. (a) The entire scene is voxelized into a volume V_η, which stores surface normals. (b) The delta volume V_Δ is initialized with a *split-inject*, where each pixel in the RSM is first identified as either real or virtual. Pixels belonging to real surfaces are left untouched. If the pixel belongs to a virtual surface, a VPL created from R_ρ and a negative VPL at the same pixel in R_μ are injected into their respective voxels. The result is identical to a differential of volumes V_ρ and V_μ. If the leftover negative contribution is sampled and added to the real background image, it effectively cancels out existing illumination, creating a shadow on a real surface. The inverse is true for the positive bounce from the virtual object. (c) A filtered chain hierarchy of each volume is created using hardware mipmapping. (d) Cone-tracing is executed in a simple ray-tracing manner, where instead of integrating multiple rays over the cone, the subtended solid angle of the cone is used as an index into the filtered volumes to lookup the correct value.

The construction of the delta-volume, detailed in Figure 5.11, is divided into four steps.

Scene Voxelization The real scene and the virtual object are voxelized into a volume V_η of 256^3 entries or more in a similar fashion to Schwarz and Seidel [SS10]. The volume has a cubical dimension twice the largest bounding box edge of O. For each rasterized fragment I save the normal of the surface and a floating point marker α to flag the voxel as occupied. This volume is later used to perform the actual cone-tracing, as it contains the entire voxelized scene and decouples operation cost from geometric complexity. As seen in Figure 5.11(a), this volume stores surface normals per voxel and is later used for visibility tests.

Split-Inject Two RSMs are created for each light source: One RSM R_μ for the real scene only and another R_ρ for both the virtual object and the real scene. In a subsequent step called *split-inject*, each pixel is identified to belong to either a real or virtual surface. If a pixel p is identified as virtual, I create two VPLs, one from $\Phi_{\rho,p} = R_\rho(p)$ and another from $\Phi_{\mu,p} = -R_\mu(p)$. Both VPLs are then injected into the volume at their respective position. Positive VPLs $\Phi_{\rho,p}$ will be placed on voxels of the virtual surface, while negative VPLs $\Phi_{\mu,p}$ end up in areas shadowed by O (see Figure 5.11(b)). Identical VPLs from both RSMs cancel each other out. The radiant intensity I of a VPL created from pixel p of an RSM with reflected flux Φ_p that I store in each voxel is:

$$I(p) = \frac{\Phi_p}{\pi} \tag{5.17}$$

To be able to compute GI for the virtual object, values $\Phi_{\rho,p}$ can be injected into a separate volume V_ρ in the same shader pass.

Filtering In Figure 5.11(c) all three volumes, collectively referred to as $\mathbf{V} = \{V_\rho, V_\Delta, V_\eta\}$, are filtered using hardware mipmapping. By filtering

the occlusion marker α in V_η into higher levels of the hierarchy, the average amount of occlusion per voxel can be queried while tracing visibility.

Cone-Tracing For each cone, I accumulate occlusion α and illumination c by ray marching through $\mathbf{V}$ along its direction, where the step size coincides with the current voxel size. After each step, $\mathbf{V}$ is queried for the filtered values α_v and c_v and both variables are updated from their previous values α_0 and c_0 in a volumetric front-to-back accumulation manner:

$$c = \alpha_0 c_0 + (1 - \alpha_0)\alpha_v c_v \tag{5.18}$$

$$\alpha = \alpha_0 + (1 - \alpha_0)\alpha_v \tag{5.19}$$

The cone-tracing operation is denoted as $\mathbf{C}(\mathbf{V})$ with a subscript to indicate the kind of samples that are gathered (e.g. $\mathbf{C}_\rho(\mathbf{V})$ samples V_η for visibility and V_ρ for indirect bounces).

5.4.2 Construction

Direct Relighting $\mathbf{T}_{\Delta_1}$ The change in direct illumination is mostly governed by shadows dropped from non-emissive virtual objects onto real surfaces. To remove existing energy where light is blocked by a virtual object, I cast a single shadow cone from a real surface point $\mathbf{p}$ in direction of a light source $\vec{\omega_i}$ using the shadow cone operator $\mathbf{C}_s \in [0, 1]$, which computes the amount of occlusion along a cone. The cone aperture γ directly depends on the area of the light source. To compute the correct amount of antiradiance for the shadow, the occlusion coefficient is negatively multiplied with the energy that would have been produced by an incident real light source L_γ. This value can be predicted with a reconstructed BRDF f_γ:

$$\mathbf{T}_{\Delta_1} L_e = \sum_i^n -\mathbf{C}_s(\mathbf{V}, \gamma, \vec{\omega_i}) L_\gamma(\mathbf{p}, -\vec{\omega_i}) f_\gamma(\mathbf{p}, \vec{\omega_i}, \vec{\omega_o}) \langle \vec{n_\mathbf{p}}, \vec{\omega_i} \rangle_+ \tag{5.20}$$

$\mathbf{T}_{\Delta_1} L_e$ is the antiradiance added to the background image, which will correctly subtract existing light to form a shadow cast by the virtual object.

Indirect Relighting $\mathbf{T}_{\Delta_2}$ Equation (5.22) below defines operator $\mathbf{T}_{\Delta_2}$: For each surface point I sum up the diffuse contribution of 9 cones [CNS+11] with a wide aperture σ in direction $\vec{\omega_i}$. I gather the specular contribution with one cone in the reflected viewing direction $\vec{\omega_r}$ with a roughness-dependent cone aperture β. In both cases I use the cone-tracing operator $\mathbf{C}_\Delta$ to gather indirect illumination from the delta-volume.

$$\begin{aligned} \mathbf{T}_{\Delta_2} L_e \quad = \quad & \sum_i^9 \mathbf{C}_\Delta(\mathbf{V}, \sigma, \vec{\omega_i}) f_\gamma(\mathbf{p}, \vec{\omega_i}, \vec{\omega_o}) \langle \vec{n_\mathbf{p}}, \vec{\omega_i} \rangle_+ + & (5.21) \\ & \mathbf{C}_\Delta(\mathbf{V}, \beta, \vec{\omega_r}) f_\gamma(\mathbf{p}, \vec{\omega_r}, \vec{\omega_o}) \langle \vec{n_\mathbf{p}}, \vec{\omega_r} \rangle_+ & (5.22) \end{aligned}$$

Because reflections of virtual surfaces block reflections of real surfaces behind them, care needs to be taken especially in cases of high specularity: Many reflections in the real image result from *unreconstructed* geometry. Simply adding the indirect bounce on top of the image will create the reflection of a translucent object rather than a fully opaque one. Because of this, the sampled visibility α from $\mathbf{C}_\Delta$ is used to weight the contribution in the background image.

5.4.3 Virtual Object Illumination

The illumination of a virtual surface follows Equations (5.23) and (5.25) below, where the indirect computation differs from Equation (5.22) only in the use of the regular cone-tracing operator $\mathbf{C}_\rho$ (to either reconstruct or directly

sample V_ρ) and a microfacet BRDF f_r with a GGX distribution [WMLT07] for specular reflections in the direct transport operator $\mathbf{T}_\rho$.

$$\mathbf{T}_\rho L_e = \sum_i^n L_\gamma(\mathbf{p}, \vec{\omega_i}) f_r(\mathbf{p}, \vec{\omega_i}, \vec{\omega_o}) \langle \vec{n_\mathbf{p}}, \vec{\omega_i} \rangle_+ \tag{5.23}$$

$$\mathbf{T}_\rho^2 L_e = \sum_i^9 \mathbf{C}_\rho(\mathbf{V}, \sigma, \vec{\omega_i}) f_r(\mathbf{p}, \vec{\omega_i}, \vec{\omega_o}) \langle \vec{n_\mathbf{p}}, \vec{\omega_i} \rangle_+ + \tag{5.24}$$

$$\mathbf{C}_\rho(\mathbf{V}, \beta, \vec{\omega_r}) f_r(\mathbf{p}, \vec{\omega_r}, \vec{\omega_o}) \langle \vec{n_\mathbf{p}}, \vec{\omega_r} \rangle_+ \tag{5.25}$$

5.4.4 Final Composition

When compositing the real scene with the rendered virtual result, I use a G-Buffer $G(p)$ which contains the reconstructed albedo of the real scene as well as the albedo of the virtual object, a binary mask with real geometry tagged as 1 and everything else as 0, and normals for each pixel p. Additionally, a buffer $B(p)$ contains the real background image. The assumption I make is that, given a faithful reconstruction of the scene, $\left(\mathbf{T}_\mu^2 + \mathbf{T}_\mu\right) L_e \approx B(p)$, i.e., the real background image can be approximated by a radiance field simplified as a volume. If this assumption is accurate, $\left(\mathbf{T}_{\Delta_2} + \mathbf{T}_{\Delta_1}\right) L_e$ should likewise faithfully correct $B(p)$ for the change in this radiance field.

For all reconstructed light sources L_γ, the virtual object is rendered the regular way by adding direct and indirect contribution of V_ρ to include bounces from virtual and real geometry on the objects surface (see Equation (5.23) and (5.25)). For all other pixels p of the background image I use operators $\mathbf{T}_{\Delta_1}$ and $\mathbf{T}_{\Delta_2}$ to compute the change in illumination which is added to the background image.

5.4.5 Implementation

I have implemented the system described in Section 5.4.1 using Direct3D 11. A Microsoft Kinect camera was used to capture the background image, and a UEye UI-2230-C camera in conjunction with a fish-eye lens to cap-

Figure 5.12: Indirect bounces, from purely diffuse to specular, are supported by DVCT. This image, rendered in 26ms, features glossy indirect reflection of the virtual Buddha in a real surface, as well as diffuse bounces from real surfaces on the Buddha.

ture surrounding real light. Marker based tracking with OpenCV [Bra00] was used to geometrically register a virtual object. I registered a manually reconstructed model of the real scene, as there is currently no feasible single-view or adaptive method to reconstruct glossy or mirroring surfaces online.

In Figure 5.12 I show mutual illumination interaction between a real-time augmentation of the Stanford Buddha and the real scene, rendered in 26ms. The scene consists of a real brushed metal surface and two diffuse reflectors in green and red. The Buddha's reflection can be seen on the metal surface next to a real wooden figurine. At the same time, the Buddha receives a red tint from reflected real light bouncing off the red diffuse surface. In this example, DVCT evaluates only the first indirect bounce, which is why the Buddha's base reflected on the metal surface has a completely black

appearance.

In Figure 5.13, an overview of the effects achieved with DVCT can be seen: After scene voxelization into 256^3 voxels (Figure 5.13(a)), soft-shadows can be faithfully reconstructed by casting a single cone to each light source (Figure 5.13(b)). Diffuse indirect bounces from virtual onto real surfaces are possible with multiple wide-angled cones (Figure 5.13(c)), while a single specular cone can be used to render mirror-like to glossy reflections (Figure 5.13(d)).

5.4.6 Error

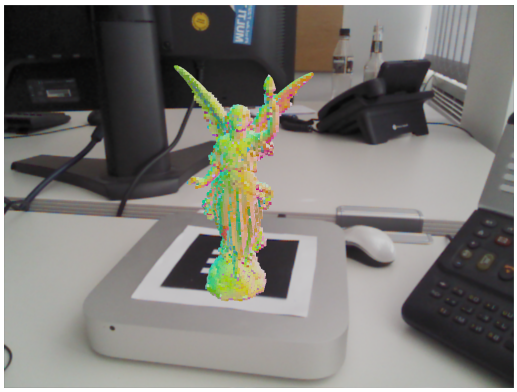
According to Equation (3.11), creating two volumes which contain indirect bounces from the real scene V_μ and additionally from both the real scene and the virtual object V_ρ, a delta-volume can be extracted which represents a corrective factor on V_μ to reconstruct V_ρ:

$$V_\rho = V_\mu + V_\Delta \tag{5.26}$$

However, this is only true for the lowest level of the prefiltered chain of all volumes. Since cone-tracing $\mathbf{C}(\mathbf{V})$ is executed on separately filtered volumes, an error E is introduced:

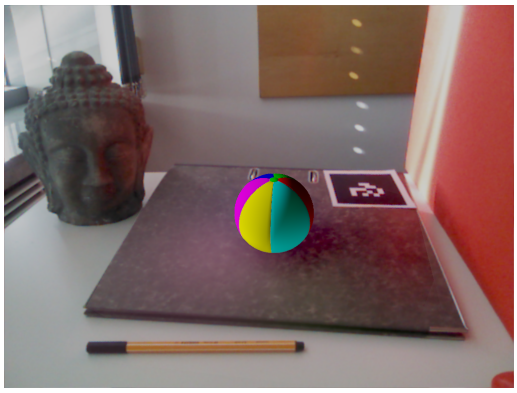
$$E = \mathbf{C}_\rho(\mathbf{V}) - [\mathbf{C}_\mu(\mathbf{V}) + \mathbf{C}_\Delta(\mathbf{V})] \tag{5.27}$$

In Figure 5.14 an error analysis shows the squared and enhanced difference $E^2 \cdot n$ between regular VCT used on a Cornell Box containing the Stanford Happy Buddha and DVCT, where the Buddha has been placed into an already lit Cornell Box. Similar to DLPVs, the voxelization of the scene might introduce bleeding artifacts, usually noticeable at edges where scene



(a) Voxelization

(b) Soft shadows



(c) Diffuse indirect bounce

(d) Glossy reflection in a real surface

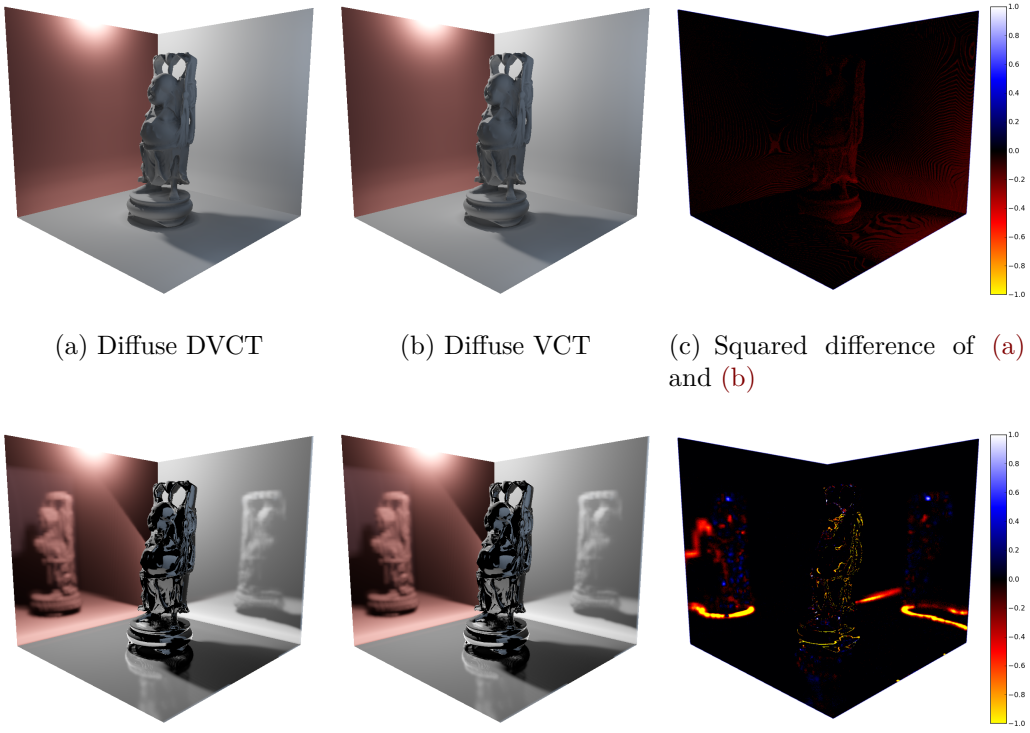
Figure 5.13: Properties of Delta Voxel Cone Tracing. (a) Virtual and real surfaces are voxelized and prefiltered. This simplified structure of the scene is later used for tracing operations and to quickly query integrals of subtended solid angles. Here the voxelized virtual object is shown. (b) The prefiltered voxelization is used to evaluate shadows with arbitrary hard edges cast from virtual objects onto real surfaces with a single cone. (c) Diffuse indirect reflections are quickly computed using multiple cones to sample the hemisphere of each surface point while in (d) glossy indirect reflections are evaluated using a single cone with an aperture matching the surface roughness. Images (b-d) were rendered at 22ms to 26ms each frame.

and augmenting object intersect. However, since the volumes are much more fine-grained, the expectation is that errors are bounded depending on the level of the filtered mipmap level. The images show that the error, depending on the specular roughness of the material, are distributed in a smaller area around the base of the object. For diffuse bounces, the difference is spread across larger areas but is less noticeable, since structural artifacts from the voxelization blur out with larger cones.

In Figure 5.15 I compared the result of DVCT with a ground truth sample rendered with Mitsuba [Jak10]. The bias introduced by the voxelization and the regular hierarchical structure of the delta-volume are a considerable source of errors. In particular, the ambient occlusion computed by using the same wide aperture for cones as for the diffuse indirect bounce cause small crevices either to darken too quickly or miss any self-occlusion entirely. Furthermore, shadows evaluated with one shadow cone may exhibit jagged edges, depending on the local resolution of the voxel grid. I currently only evaluate the first indirect bounce, which can have drastic effects in scenes with many glossy or specular surfaces, as can be seen in Figures 5.15(c) and 5.15(e): Multiple bounces lead to considerably more energy in the scene. For better comparison, I show two images generated with Mitsuba, once with only one indirect bounce in 5.15(d), and once with arbitrary path lengths 5.15(e).

5.4.7 Performance

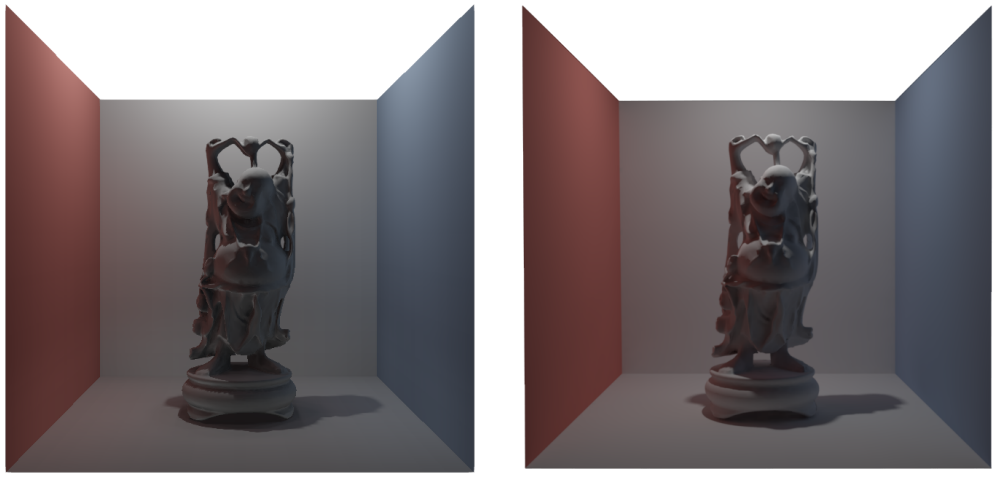
I have gathered timings for the entire pipeline on a test system with an Intel i7 X980 and a NVIDIA GTX 780. A measurement of different scenes is shown in Table 5.6, where each step of the pipeline is listed. Gray rows represent steps necessary to create and evaluate a DVCT. The setup of these measurements consists of RSMs rendered at 1024^2 resolution, where all pixels were used to generate and inject VPLs into the volumetric textures V_ρ and V_Δ. The general resolution of $\mathbf{V}$ was 256^3 voxels and the resolution of the video stream was 720p. Cone-tracing is performed after a deferred renderer creates a G-Buffer for the scene, where virtual and real reconstructed geometry are combined. An additional post-processing pipeline includes a step for mor-



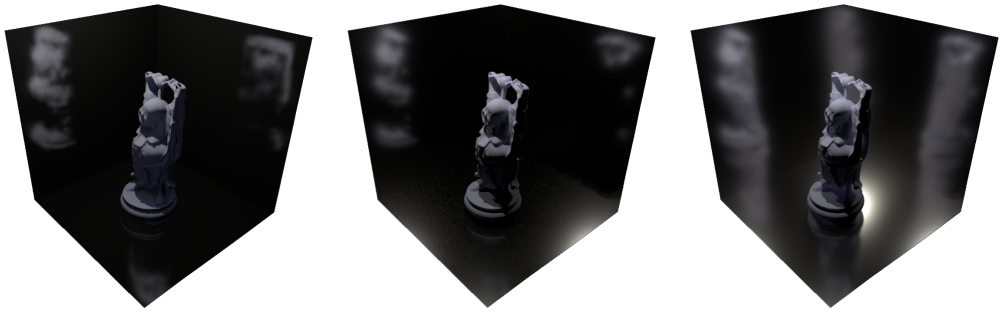
(a) Diffuse DVCT (b) Diffuse VCT (c) Squared difference of (a) and (b)

(d) Specular DVCT (e) Specular VCT (f) Squared difference of (d) and (e)

Figure 5.14: Error analysis (negative errors in yellow-red, positive errors in blue-white). (a) The Stanford Buddha is augmented into an already lit Cornell Box with DVCT. (b) The same scene rendered with regular VCT, i.e., one volume for the entire scene. (c) The squared difference of both images (enhanced with $n = 16$). Compared to regular VCT, DVCT subtracts slightly too much energy. The indirect bounce from the blue wall to the left of the Buddha (not visible from viewers perspective) is dampened too much by the antiradiance. Images (d), (e) and (f) show the same sequence with glossy surfaces. As expected, the error is now concentrated along the edges in the reflection, most notably the edge of the shadow. The front of the Buddha has a few spots which are slightly too bright, but this error is not as apparent as the overcompensation due to antiradiance.



(a) Diffuse DVCT (b) Diffuse ground truth



(c) Glossy DVCT (d) One indirect bounce (e) Glossy ground truth

Figure 5.15: Comparison with ground truth. (a) The Stanford Buddha is augmented into an already lit Cornell Box with DVCT rendered in 26ms. Shadows are evaluated with one narrow shadow cone per pixel. Cavities on the surface of the Buddha appear too dark in comparison to (b) the result rendered with Mitsuba [Jak10] and 1024spp with a Primary Sample Space Metropolis Light Transport integrator in Mitsuba after 28 minutes. Figure (c-e) show the same scene rendered with a rough conductor material. Figure (c) was rendered in 37ms with DVCT. For better comparison Figure (d), rendered with 8192spp in 2 hours, shows the ground-truth with fixed length light-paths of 3, while Figure (e) shows ground-truth with reflections of arbitrary path-length was rendered in 4.1 hours.

Pipeline step	Infinite Head	Phlegmatic Dragon	Happy Buddha
R_μ	0.19	0.22	0.18
R_ρ	0.20	0.37	0.62
Voxelization	2.29	3.04	4.39
Split-Inject	3.38	3.45	3.26
Filtering	7.90	8.03	7.68
Deferred pass	0.24	0.40	0.64
Postprocessing	0.68	0.74	0.65
Δ Cone-Tracing Indirect + Shadow			
Diffuse	7.21	10.36	9.07
Glossy	8.84	10.65	10.30
Specular	20.00	21.17	30.53
$\sum$	23.72	26.90	27.72
DVCT Cost	22.80	25.76	26.43

Table 5.6: Detailed timings per frame in milliseconds taken for each step of the pipeline, with 1024^2 VPL injections for a volume $\mathbf{V}$ of size 256^3 rendered in 720p. The highlighted rows display the effective time to render the indirect contribution and soft shadows. Because cone-tracing performance varies with opening aperture, I gathered numbers for three types of surface roughness. In the last two rows the sum of these operations for glossy indirect reflections on both real and virtual surfaces are added up. The computation time for DVCT is roughly 25ms on average.

phological antialiasing, HDR tone-mapping and gamma correction. DVCT differs from regular VCT in the time for the split-injection and its subsequent filtering, and in case V_ρ is used to shade the virtual object in the additional memory requirement for another volume.

In Table 5.7 I have varied the sizes of the volume used to represent the scene and show their creation cost, as well as the variation of the time spent to evaluate a single pixel with 9 cones for diffuse and 1 cone for specular indirect light. Since my current implementation does not make use of spatial data structures to compress the volume, the cost is linearly dependent on the resolution of the volume.

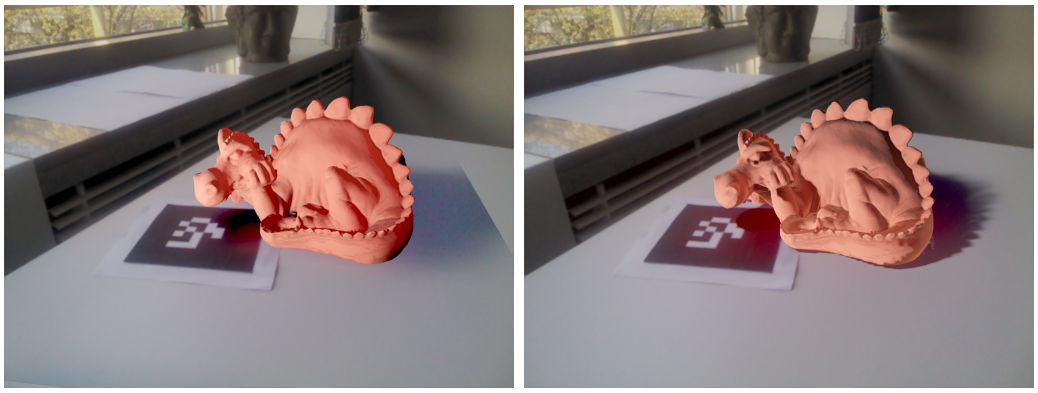
DVCT steps	V Resolution			
	64^3	128^3	256^3	512^3
Voxelization	0.95	1.27	3.03	17.93
Split-Inject	1.19	1.45	3.29	205.66
Filtering	2.32	6.07	7.79	251.76
Cone-Tracing	3.47	5.33	8.70	24.47

Table 5.7: Time spent on each step of a DVCT pipeline with varying volume resolution in milliseconds.

5.4.8 Evaluation

In DLPVs from Section 5.3 the difference of two RSMs R_ρ and R_μ is injected and propagated to relight the real scene, and a regular LPV is constructed to add indirect bounces to the virtual object. DLPVs lose most of their directional information due to the low second-order spherical harmonic encoding and therefore provide only diffuse indirect reflections. Because the volume is injected with a direct component as well, the same volume will also naturally encode shadowed areas from direct light sources due to direct antiradiance. These areas however suffer from considerable aliasing artifacts due to the low resolution of the volume, which is typically in the range of 32^3 to 64^3 voxels. Shadows can therefore only have very soft appearances. The same low resolution causes bleeding artifacts through thin geometry visible both as too much additional light or wrong coloring due to subtraction from antiradiance.

In Figure 5.16 a side by side comparison shows that the DVCT approach yields significantly better shadow resolution, has fewer bleeding artifacts and is not bound by either the dimension of the volume (the shadow in the DLPV gets cut off) or the propagation distance. DVCT can also simulate a bandwidth of mutual indirect reflections from diffuse, glossy or specular surfaces, which is not possible with DLPVs. DLPVs however outperform DVCT for diffuse bounces, as they operate on smaller volumes. Surprisingly, much time is spent on the three injection steps to V_ρ and the two delta injects into V_Δ in the DLPV.



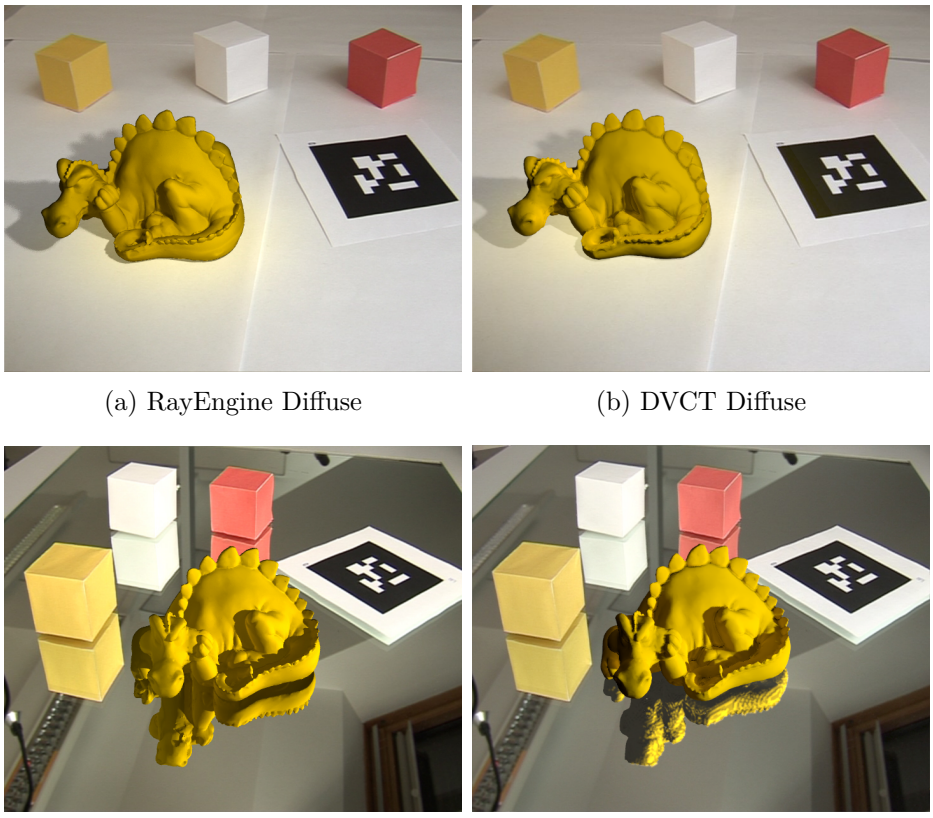
(a) DLPV (b) DVCT

Figure 5.16: DLPV comparison using 1024^2 VPLs. (a) DLPV with 32^3 voxels in 22ms. (b) DVCT with 256^3 voxels in 26ms. Both volumes have the same dimensions.

DLPVs propagate illumination in a volume which can only be queried inside its boundaries. As a consequence this can lead to shadows and bright indirect bounces being cut off at the edge of the volume. Because DVCT is a gathering process, an improvement over DLPVs is that there is no distance restriction for indirect illumination and shadows (see Figure 5.16).

Augmented specular reflections, refractions and caustics on real surfaces have been presented by Kán and Kaufmann [KK12] using a GPU implementation of photon mapping (RRC). In a follow up publication [KK13], the work has been enhanced with Differential Irradiance Caching (DIC) to simulate diffuse reflections. The combination of both methods is referred to as RayEngine [KK10]. While the ray-tracing architecture naturally provides specular reflections on mirror-like surfaces, diffuse indirect reflections need dense sampling to appear noise-free or have to rely on caches which lead to costly updates whenever they are out of date. Because of the choice for the current architecture, there is no support for transitional reflection types on glossy surfaces.

Figure 5.17 shows a comparison between DVCT and RayEngine of two scenes with diffuse indirect and specular indirect reflections using the same settings and a single light source. For images rendered with RayEngine, dynamic recalculation of the irradiance cache was turned for dynamic scenes and was



(a) RayEngine Diffuse (b) DVCT Diffuse

(c) RayEngine Specular (d) DVCT Specular

Figure 5.17: Comparison to RayEngine. Upper row: Diffuse indirect reflections of (a) DVCT in 33ms and (b) DIC in 714ms. Lower row: Specular indirect reflections of (c) DVCT in 28ms and (d) RRC in 588ms. All images were rendered with a resolution of 720 × 576 pixels. DVCT computes comparable effects at $\frac{1}{20}$ of the cost.

generated on an NVIDIA GTX 690. The computation time for Figure 5.17(a) was 714ms versus 29ms for 5.17(b), and 588ms for 5.17(c) versus 28ms for 5.17(d).

Because DVCT relies on a voxelized scene for cone-tracing, structural artifacts appear in the reflection in Figure 5.17(d). These can be removed either by further filtering or with higher volume resolution, but lag behind ray tracing, as there is currently no measure to construct arbitrarily fine-grained volumes to capture similarly detailed geometry. Currently DVCT supports only single bounce indirect reflections, which is why the reflection shows the unlit side of the dragon completely black. Diffuse indirect reflections however are reconstructed with the same fidelity, and all types of indirect reflections

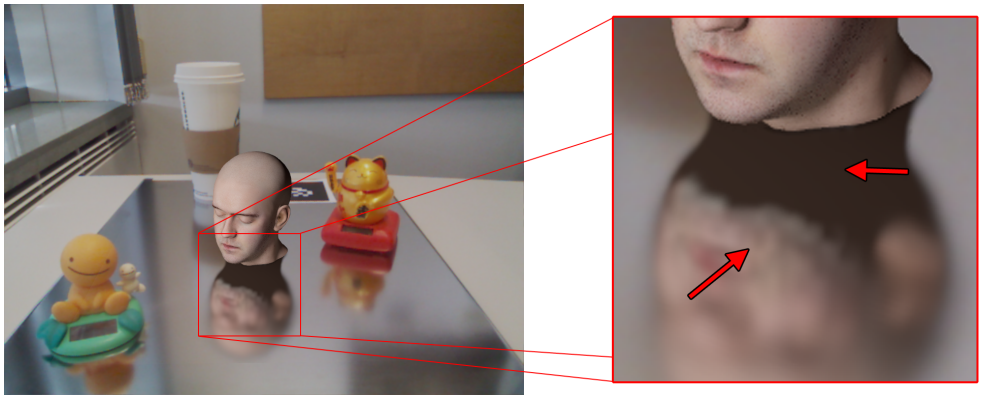


Figure 5.18: Even though the volumetric resolution is much higher than in DLPVs, structural artifacts appear in glossy reflections. Furthermore, as discussed in Section 2.2.4 missing secondary indirect bounces become more emphasized with lower glossiness.

are computed at a fraction of the cost of RayEngine.

5.4.9 Discussion

The visual fidelity and cost of DVCT is tied to the volumetric resolution of the scene. Smaller volumes have drastically lower memory requirements, but cone-tracing on such volumes may lead to aliasing artifacts, visible in Figure 5.18, and stepping through thin walls not properly sampled by the volume. Consequently, the latter artifacts cause bleeding of illumination and antiradiance.

Another category of errors is dependent on the aperture of cone angles: Wider cones can show artifacts when gathering illumination, but narrow cones are usually traced for longer paths and therefore consume more time. Wider cones can also get stuck in narrow slits or self-intersect with the scene when cast at narrow angles. In practice, I have not noticed this effect for diffuse illumination, but for conductors and dielectric materials, where evaluating view dependent illumination is done with a single additional cone, this effect can have visual impact. Tracing shadow cones on V_η which self-intersect with geometry can cause slight darkening due to precision issues when combining antiradiance with the background image.

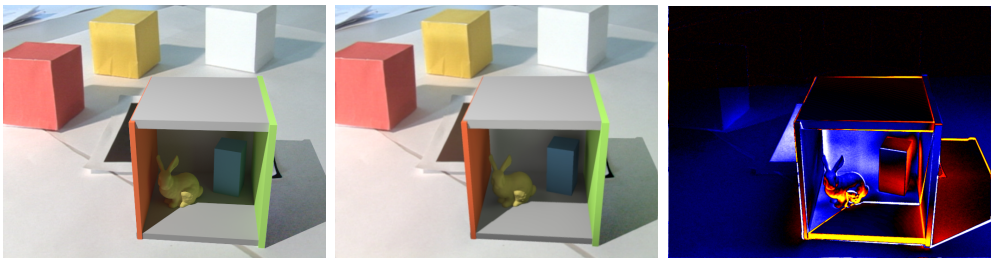
The delta-volume $\mathbf{V}$ covers the near surrounding of an object O, but can be traced from arbitrary distances for each point of the real reconstructed scene. Indirect shadows however are projected and stored as antiradiance inside V_Δ and are limited by the dimension of the volume. For instance, a shadow of O reflected in a real mirror can still be cut off at the edge of the volume, while directly tracing the occlusion from V_η works for all points of the reconstructed real scene. Similarly, using V_ρ to compute GI for the virtual object O is still subject to limited bounce range: The volume around O can only capture bounces within its boundaries. Indirect illumination on a virtual surface can change abruptly if a real reflecting surface is moving out of range.

5.5 Discussion

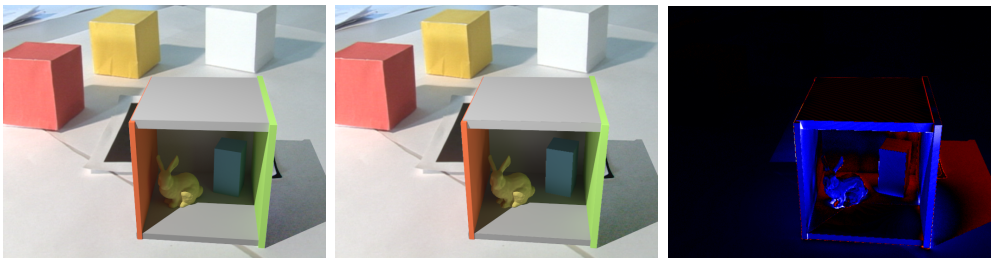
To put each method into a larger context, this section will show a grand comparison of the current State-of-the-Art in global illumination relighting and shading. Augmented Reality has two opposing goals with very tight requirements: High physical realism for seamless integration of virtual objects into real scenes and real-time execution. Each presented method comes with restrictions and trade-offs. These do not necessarily have negative consequences, as the environment and conditions in which an Augmented Reality simulation operates dictate what can and cannot be done.

In Figure 5.19, Differential Instant Radiosity [KTM+10], Differential Irradiance Caching [KK13], Path tracing with a 100ms budget, Delta Light Propagation Volumes and Delta Voxel Cone Tracing are all compared to a ground truth rendition of an augmented scene[3]. Each row demonstrates the method itself in the middle with the ground truth shown to the left, and the squared error enhanced by 4x to the right. Blue-white spectra show positive errors (i.e., the rendered image is too bright in this area), whereas red-yellow spectra show negative errors (i.e., the rendered image is too dark in this area). Furthermore, the Peak-Signal-To-Noise-Ratio (PSNR) is displayed for each row.

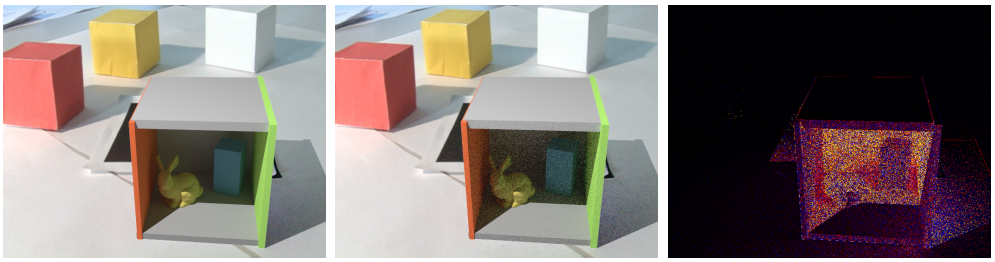
[3]This is an enhanced version of the original comparison in [KK13].



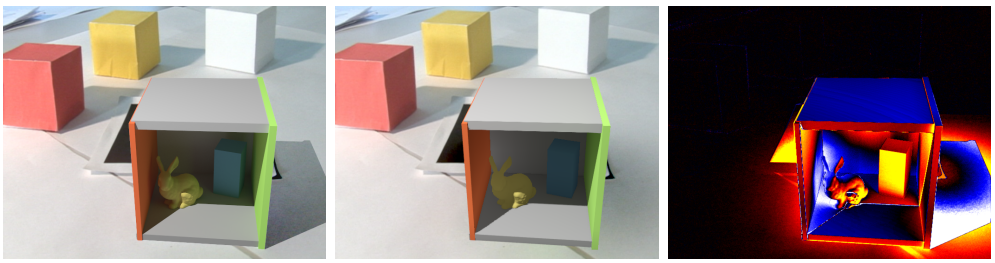
(a) DIR, 36ms (PSNR: 27.8689dB)



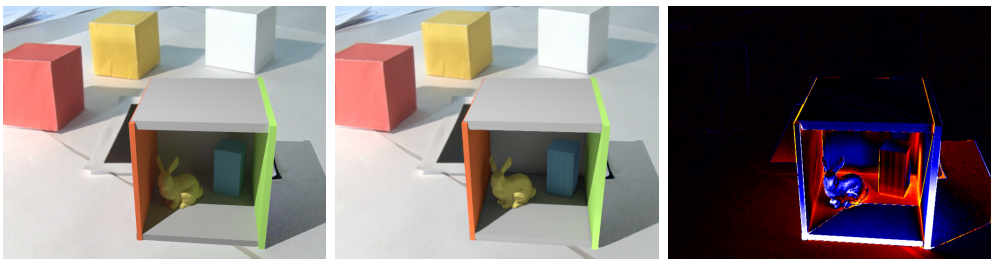
(b) DIC, 100ms (PSNR: 35.4106dB)



(c) Path Tracing, 100ms (PSNR: 31.1039dB)



(d) DLPV, 22ms (PSNR: 26.5559dB)



(e) DVCT, 28ms (PSNR: 30.2468dB)

Figure 5.19: Ground truth comparison.

The PSNR is given by:

$$PSNR = 10 \log_{10} \left(\frac{R_I^2}{MSE} \right) \tag{5.28}$$

R_I is the maximum possible pixel value in the image I and MSE the mean squared error. The first observation is that the PSNR, although a good guide for image quality in general, fails to account for sensitivities in the human visual system. As discussed in Section 2.2.4 and identified in [HFJS09], noisy images are perceived to be less realistic than biased images with a much lower PSNR with limited global transport. This is also true for the Figure 5.19(c): Even though the PSNR is higher than almost all other real-time methods, this result is inadequate for simulation purposes.

Delta Light Propagation Volumes in Figure 5.19(d) score lowest and also show the highest amount of *hot-zones* in the error images. These are particularly distinctive in the area of the shadow, as the low resolution of the the DLPV volume and the low-frequency encoding cannot capture the sharp outline of the shadowed area and simply creates a large circular blob behind the box. Another particular downside to the DLPV/LPV combination is that they are not very well suited for objects which contain lots of internal features such as this Cornell box. Most of the light inside the box is blurred or bleeding into other regions, distributing light more regularly across the box and therefore missing out indirect shadowing. This is visible in the area behind the bunny, which did not occlude the indirect light coming from the front and therefore being too bright. The reverse situation is visible almost everywhere else. On top of the box, the self-illumination issue with DLPVs/LPVs becomes readily apparent: The surface is slightly brighter than it should be. This area is interesting because it does not receive much indirect light from anywhere else and shows no artifacts in all other methods.

Differential Instant Radiosity in Figure 5.19(a) employs Imperfect Shadow Maps [RGK+08] to add indirect shadows to regular Instant Radiosity through Reflective Shadow Maps. This is clearly visible, as the shadow to the left of

the bunny for instance is much less attenuated in the error image. Of particular interest are surfaces near their indirect reflectors: The bunny appears too dark on its front side, yet receives too much indirect energy when not surrounded by nearby geometry (see for instance its top side). Very similar is the edge of the Cornell box touching the ground, or the slight darkening next to the Cornell box on the left. Imperfect Shadow Maps tend to underestimate the indirect lighting in these areas and overestimate them far away, as is visible in the large surrounding excess bounce.

Differential Irradiance Caching in Figure 5.19(b) shows the least amount of errors, which are spread in an almost uniform fashion: Light-facing sides of objects tend to be overestimated from the cache, while surfaces facing away from either direct or indirect light tend to be underestimated. The most severe version of this error can be seen at the base of the bunny and other small crevices, where front and back-facing sides of the foot of the statue receive too much and too little energy accordingly. Differential Irradiance Caching also features secondary indirect bounces, which reduce the error on the right side of the blue box, which is partially lit by the indirect green bounce of the wall, and the left side of the bunny, which receives a orange tint from the wall to its left.

The main contender to this level of quality is therefore Delta Voxel Cone Tracing, shown in Figure 5.19(e). As expected, the voxelization will lead to bleeding or aliasing artifacts near edges, which generally appear too dark. Aliasing can be seen on the blue box, which shows a pattern of unequal indirect bounces. Indirect shadows are however preserved better than in the case of DLPVs and DIR. The highest amount of errors is visible on the edges of the box, which gather too much light from the ground.

Given this error analysis, it is important to identify shortcomings of each method to determine which option may be suitable under which circumstances. In this scene for instance, a dominant effect are secondary indirect bounces clearly visible in the scene that are not accounted for in most methods. These however only appear because of the structure of the object. When augmenting a scene with a more convex object such as a single round bust, this factor tends to be less of a problem than it is here.

	Real-Time	Temporal Coherency	GPU Friendliness	Diffuse Bounces	Glossy Bounces	Specular Bounces	Volumetric Effects
DVCT [Fra14a]	★★★★☆	★★★★★	★★★☆☆	★★★★★	★★★★★	★★★★☆	★★☆☆☆
DLPV [Fra13a]	★★★★★	★★★★★	★★★★☆	★★★★☆	☆☆☆☆☆	☆☆☆☆☆	★★★☆☆
RRC [KK12]	★★★☆☆	★★★★☆	★★★★☆	☆☆☆☆☆	☆☆☆☆☆	★★★★★	★☆☆☆☆
DIC [KK13]	★★☆☆☆	★★★☆☆	★★★★☆	★★★★★	☆☆☆☆☆	☆☆☆☆☆	★☆☆☆☆
MRS [LB12]	★★★★★	★★★☆☆	★★★★★	★★★★☆	☆☆☆☆☆	☆☆☆☆☆	☆☆☆☆☆
LightSkin [LB13c]	★★★★★	★★★★☆	★★★★★	★★★★★	★★★☆☆	☆☆☆☆☆	★★★☆☆
AR-OOF [FKOJ11]	★★★☆☆	★★★★☆	★★★★☆	★★★☆☆	☆☆☆☆☆	☆☆☆☆☆	☆☆☆☆☆
D-RA [RBDG14]	★★★☆☆	★★★★☆	★★★☆☆	☆☆☆☆☆	☆☆☆☆☆	☆☆☆☆☆	☆☆☆☆☆
DIR [KTM+10]	★★★☆☆	★★★☆☆	★★★★☆	★★★★☆	★★☆☆☆	☆☆☆☆☆	☆☆☆☆☆

Table 5.8: Comparison of different relighting methods. The table rates different aspects of each method. DVCT is a clear compromise between accuracy and efficiency.

In Table 5.8, all current publications which deal with global illumination in AR are listed with their respective strengths and weaknesses. Next to real-time capability, an important factor for choosing one method over another are supported effects and temporal coherency.

Temporal coherency is overall important as it is connected to the same visual perception issue regarding high-frequency noise: Inconsistencies from one frame to another are quickly recognized and perceived as unrealistic. In scenes with smoothly changing lighting conditions over time and changing camera positions lack of temporally coherent rendering is unforgiving.

In the special case where the real scene contains particles or other media which need to be addressed, volumetric methods and LightSkin offer good support. Because there are currently no good methods to reconstruct these real effects in order to relight them, application in this are remains very limited.

Scenes which contain metal objects or highly reflective surfaces will quickly reveal virtual objects if they are not properly relit to match the reflection of the virtual object. In these cases, only two methods – RayEngine (DIC and RRC combined) and DVCT – stand out, with only one of them supporting glossy reflections. This is because while LightSkin and DIR are technically able to reproduce glossy reflections, LightSkin cannot be used for relighting other surfaces (as the authors concede in the publication [LB13c]) and DIR would require many VPLs, exceeding the real-time budget.

While RayEngine reproduces perfect specular reflections and diffuse bounces much more accurately, the very high price in rendering time in a completely dynamic scene and the missing support for glossy reflections force the conclusion that DVCT is the currently most versatile shading and relighting method for AR, while reproducing the scene at a good estimate to work with. With growing hardware capabilities and better support for voxelized scene abstractions, it may also become much better at handling small-scale artifacts and reducing over-/underestimation from filtering.

5.6 Conclusion

In this chapter I have formulated three novel methods, built on the theory of Delta Radiance Fields, to relight real surfaces or transfer effects a virtual object has on a real environment. This chapter has addressed and solved **Problem 2** and **Problem 3** discussed in Section 1.1. The following three contributions were presented to do so: AR-OOFs, DLPVs, and DVCT.

AR-OOFs AR-OOFs are volumetric entities which precompute the occlusion of a virtual object from a viewing direction on one or more spheres around it. Several such shells are set up to create a spherical field L_Δ of occlusion coefficients, which are later convolved with the transfer coefficients of a real surface using a triple product. The resulting coefficients represent the surface transfer combined with the new visibility. A second convolution of this refined transfer yields the relit result. AR-OOFs can be trivially expanded to support indirect bounces of light. However, because of their high computational demand and their low-frequency encoding, they are not to be preferred over other methods which deliver better renditions of the same phenomena at much lower cost, such as DLPVs.

DLPVs and DVCT Delta Light Propagation Volumes and Delta Voxel Cone Tracing are two volumetric methods which inject and store the delta between two RSMs into a volume. DLPVs propagate this delta through the volume by scattering the contribution to neighboring voxels while DVCT uses cone tracing to collect the injected delta. DLPVs are faster than DVCT, but suffer from large bleeding artifacts which stems from the low resolution of the volume necessary to ensure high performance of propagation. Because DVCT relies on a gathering scheme, the propagation step is effectively cut out and replaced by a method which has the ability to compute not only diffuse, but glossy and specular reflections. This is important for scenes where a virtual object should be reflected on a glossy or highly specular surface.

I have compared all three algorithms with several other methods in the scientific literature [KK13, LB12, LB13c, RBDG14, KTM+10]. Obviously different requirements regarding the system (such as mobile support) lead to different emphasis on rendering aspects. Nevertheless, the latest development – Delta Voxel Cone Tracing – offers significant advances when compared to competing algorithms: It is fast enough to be considered real-time, simulates diffuse and rudimentary perfect specular bounces and is currently the only method to also include glossy indirect reflections for real-time AR relighting. The same method is also able to cast arbitrary soft shadows from the virtual object onto real surfaces. Furthermore, this chapter has demonstrated that DVCT, the current iteration of volumetric methods I have investigated, also reduces the error to the ground truth compared to other non-path-tracing options significantly while avoiding artifacts which negatively impact perceived realism.

DVCT comes at the price of high bandwidth and memory costs for the volume. Simulating multiple objects requires multiple volumes, which is unfeasible for devices with low memory capacities such as mobile devices. In this instance, Delta Light Propagation Volumes serve as a stable replacement. Though they cannot be used to simulate glossy or highly specular reflections, much lower requirements in bandwidth and the fast propagation time play in their favor when simulating convex objects with diffuse surrounding materials. DLPVs do not suffer from temporal inconsistencies as other methods do and their low-frequency range is well suited for scenarios where either scene conditions are ambient (i.e., real materials and lighting conditions are very diffuse) or unknown (in this case simulation of ambient conditions is the preferred choice, much as Ambient Occlusion is statistically correct for unknown lighting conditions).

DLPVs and DVCT efficiently solve the problem of relighting real surfaces as laid out in Section 1.1 and improve the current state of the art significantly.

Chapter 6

Conclusion

6.1 Summary of Contributions

Augmented Reality shading and relighting is a notoriously difficult problem in rendering, as the human visual system is very sensible to discrepancies of

Figure 6.1: Reality versus Augmentation. A 3D printed model of the XYZRGB Dragon in comparison to an augmentation on the right.

virtual conditions when directly exposed to their real counterparts. Reconstruction of real surfaces and materials need to match their physical counterparts as closely as possible in order to minimize these discrepancies, but this is only one part of the source of errors. The reconstruction and simulation of global illumination is the key component to create a truly *mixed* reality.

The proper rendition of a virtual object within a real existing context is faced by three major problems: The *shading* of the objects surface according to real illumination from its surrounding, the *relighting* of the surrounding to match the new conditions imposed by the presence of a new object, and the consistent *global interaction* of light between both worlds. Contemporary methods rely on Differential Rendering techniques to extract illumination differences between augmented and unaugmented scenes. This approach is simple and unobtrusive to the underlying shading mechanisms, but tends to be expensive once the cost of said shading algorithms rises to a real-time threshold, preventing the use of modern, highly realistic global illumination techniques in AR. A byproduct is the reduced bandwidth of light-material interaction that these methods can simulate, often only reproducing low-frequency bounces on virtual and real surfaces. To attack these problems, I have split the problem of photometric registration of virtual objects into

shading and relighting to propose individual solutions best suited for different scenarios.

In this thesis I have presented a mathematical framework to describe and construct delta transfer operators, replacing regular Differential Rendering for AR. Building on delta transfer operators can significantly reduce the workload of augmenting an image with a new object and its exerted influence on the lighting conditions of its surrounding. This is an important goal when optimizing global illumination algorithms for Augmented Reality simulations to achieve real-time behavior, where I defined real-time to have an upper cost of 33ms per frame.

I have conducted several experiments with real-time global illumination shading algorithms and proposed two new methods to render an object – either entirely rigid with static materials or completely dynamic – with natural illumination from a real environment. The outcome of these experiments yield a solution to shade virtual objects according to their surrounding conditions, effectively solving the shading problem discussed in Section 1.1. These two methods yield to different scenarios such as for instance online car configurators, which can change an object arbitrarily, or the reproduction of real objects, such as museum artifacts with static properties.

I have proposed three novel methods to relight an environment with volumetric methods. In comparison to contemporary algorithms in the scientific literature, these methods provide stable, temporally coherent illumination with very high efficiency and the last iteration is able to simulate effects currently not demonstrated anywhere else. The result of the conducted analysis is important when incorporating relighting algorithms into an Augmented Reality renderer, because differences in appearance and realism are sometimes scene dependent. It can therefore be beneficial to prefer an overall less realistic method for better efficiency or mobile support. Overall however, Delta Voxel Cone Tracing is arguably the best trade-off between efficiency, visual effects and bias. Because of the important role of noise in the perceived realism of an augmented image, the only contender in image fidelity, i.e., unbiased path-tracing capped to a frame budget of 100ms, cannot be used to create a visually appealing augmentation, and Differential Irradiance

Caching is far from matching real-time efficiency. In light of these reasons the presented methods of Chapter 5 provide a qualitative and efficient solution to the relighting problem discussed in Section 1.1 and 5.1.1 and advance the current state of the art.

The algorithms presented in this thesis may also be used for virtual relighting. Many rendering tasks, for instance in computer games, often only require single objects to gather and exert influence while the rest of the scene remains static. In these cases relighting methods presented in Chapter 5 can augment precomputed illumination with the additional changes.

6.2 Future Work

Real-time relighting is still a largely unexplored area. Although many lessons can be learned from the *offline relighting community* (i.e. augmenting legacy photographs or movies with path-traced solutions), many issues cannot simply be transferred to the real-time domain. The following list may provide a starting point for the interested reader:

Postprocessing Especially smaller cameras like webcams or the Microsoft Kinect often have low image quality, exhibiting artifacts such as noise, general blurriness, reduced or quantized color spectra, motion blur, lens distortion and more. Apart from missing photometric registration, true Augmented Reality immersion often suffers from the disparity in quality between the virtual object and the background image. There is currently no standard process by which one can extract parameters for all kinds of artifacts of a camera and use them in a postprocesses which will correctly degrade the rendered virtual object to match the background image. Several publications already discuss visual fidelity in terms of these artifacts [KM08, KSF10, KTPW11, OHSG12, PLW12].

Global Illumination Real-time GI in AR is closely tied to solutions presented in the general gaming environment. Because most AR applications

rely on absolute flexibility and dynamic behavior, adapting algorithms might end up challenging. Apart from the volumetric methods implemented here or PRT-based solutions for rigid objects, tiled-based rendering can increase the amount of VPLs in Instant Radiosity solutions to suppress flickering [OA11]. Furthermore, just as game artists usually tag objects as static or dynamic, rigid objects in a real scene could rely on different GI paths than dynamic objects. Screen space methods have yet to be properly investigated to enhance the image with small scale effects and local reflections.

Perceptual Rendering Since virtual objects are exposed directly for comparison to the real environment, one might tend to go overboard with rendering the object as physically correct as possible at the expense of GPU time. Here several psycho-visual experiments could further deduce important cues the human visual system relies on, and artifacts which are easily overlooked to cut down on shading costs. Because some artifacts, even though subtle, may have a huge impact on the perceived realism of a scene, they provide a guideline when developing new algorithms.

Material Reconstruction Online material reconstruction, especially for non-diffuse parameters, is a large open topic. While there are several methods to faithfully reconstruct material parameters with complicated dome setups (the MERL BRDF Database for instance [Lab06]), quickly evaluating these from a single image in a couple of milliseconds remains an elusive goal. This may not be necessary in static scene setups with known properties, but as AR technology becomes more widespread, the reliance of relighting algorithms on proper scene reconstruction for unknown environments will increasingly become vital.

New display technologies Most AR applications still use the *window-to-the-world*. In this type of simulation realism is increased with better rendering technology. However, the *immersiveness* of the application is limited severely by an essentially 2D context. AR-Rift setups, see-through contact lenses and other display technologies are important factors to create a complete approach to realistic Augmented Reality.

6.3 Closing Remarks

Augmented Reality has a bright future. Leading the charge in research is the entertainment industry, followed closely by virtual production and previsualization in sales and marketing. To me the most interesting field is the preservation of cultural heritage objects and the museum of the world of tomorrow. All of these fields share one common goal: To bring virtual objects in another context to life with – in the spirit of Étienne-Gaspard Robert's *Phantasmagoria* – an uncanny degree of realism. In the same manner as his experiments revealed the conditions in which his audience could not tell apart reality from theater, today psycho-visual analysis of several phenomena in real world stimuli may help reduce the amount of necessary visual computation in a Mixed Reality simulation. Given a visual Turing test, the precise conditions under which an object is *perceived* to be real can be determined and the last obstacle to the real-time realism overcome.

When these conditions have been satisfied, then – to the human mind – simulation and reality are indistinguishable.

Source Code

An implementation of several algorithms presented in this dissertation is available at https://github.com/thefranke. It is licensed under the MIT free software license.

```
The Dirtchamber

Copyright (c) 2014 Tobias Alexander Franke
http://www.tobias-franke.eu

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in
all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
THE SOFTWARE.
```

Curriculum Vitae

Vita

2008 – 2015	Research Associate at *Fraunhofer IGD* (Fraunhofer Institute for Computer Graphics Research), Darmstadt, Germany. Departments *Virtual and Augmented Reality* (2008 – 2010) and *Visual Computing System Technologies* (since 2010).
2007 – 2008	Graphics Researcher at *GRIS TU-Darmstadt* (Graphisch Interaktive Systene), Darmstadt, Germany.
2004 – 2006	Student of Computer Science at *Technische Universität Darmstadt*. Specialization: *Computer Graphics and Visualization*. Graduated as *Master of Science*.
2001 – 2004	Student of Computer Science at *FH Darmstadt*. Specialization: *Computer Graphics and Visualization*. Graduated as *Bachelor of Science in Computer Science*.

Conference Proceedings

2014

[Fra14a] Tobias Alexander Franke. Delta voxel cone tracing. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, pages 39–44, Sept 2014

[Fra14b] Tobias Alexander Franke. Interactive relighting of arbitrary rough surfaces. In *ACM SIGGRAPH 2014 Posters*, SIGGRAPH '14, pages 28:1–28:1, New York, NY, USA, 2014. ACM

[HF14] Lukas Hermanns and Tobias Alexander Franke. Screen space cone tracing for glossy reflections. In *ACM SIGGRAPH 2014 Posters*, SIGGRAPH '14, pages 102:1–102:1, New York, NY, USA, 2014. ACM

[OFR14] Manuel Olbrich, Tobias Alexander Franke, and Pavel Rojtberg. Remote visual tracking for the (mobile) web. In *Proceedings of the Nineteenth International ACM Conference on 3D Web Technologies*, Web3D '14, pages 27–33, New York, NY, USA, 2014. ACM

2013

[Fra13a] Tobias Alexander Franke. Delta light propagation volumes for mixed reality. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 125–132, Oct 2013

[Fra13b] Tobias Alexander Franke. Near-field illumination for mixed reality with delta radiance fields. In *ACM SIGGRAPH 2013 Posters*, SIGGRAPH '13, pages 76:1–76:1, New York, NY, USA, 2013. ACM

[WFKO13] Sabine Webel, Tobias Alexander Franke, Jens Keil, and Manuel Olbrich. Immersive experience of current and ancient reconstructed cultural attractions . In *Digital Heritage International Congress - Track 2 - - Visualization & Interaction*, pages 395–398. Eurographics Association, 2013

[OFKH13] Manuel Olbrich, Tobias Alexander Franke, Jens Keil, and Sven Hertling. Skeletal input for user interaction in x3d. In *Proceedings of the 18th International Conference on 3D Web Technology*, Web3D '13, pages 143–146, New York, NY, USA, 2013. ACM

[FSBR13] Tobias Alexander Franke, Volker Settgast, Johannes Behr, and Bruno Raffin. Vcore: a web resource oriented architecture for efficient data exchange. In *Proceedings of the 18th International Conference on 3D Web*

Technology, Web3D '13, pages 71–78, New York, NY, USA, 2013. ACM

2012

[FF12] Tobias Alexander Franke and Dieter W. Fellner. A scalable framework for image-based material representations. In *Proceedings of the 17th International Conference on 3D Web Technology*, Web3D '12, pages 83–91, New York, NY, USA, 2012. ACM

[FOF12] Tobias Alexander Franke, Manuel Olbrich, and Dieter W. Fellner. A flexible approach to gesture recognition and interaction in x3d. In *Proceedings of the 17th International Conference on 3D Web Technology*, Web3D '12, pages 171–174, New York, NY, USA, 2012. ACM

[BJFS12] Johannes Behr, Yvonne Jung, Tobias Alexander Franke, and Timo Sturm. Using images and explicit binary container for efficient and incremental delivery of declarative 3d scenes on the web. In *Proceedings of the 17th International Conference on 3D Web Technology*, Web3D '12, pages 17–25, New York, NY, USA, 2012. ACM

2011

[FKOJ11] Tobias Alexander Franke, Svenja Kahn, Manuel Olbrich, and Yvonne Jung. Enhancing realism of mixed reality applications through real-time depth-imaging devices in x3d. In *Proceedings of the 16th International Conference on 3D Web Technology*, Web3D '11, pages 71–79, New York, NY, USA, 2011. ACM

2010

[SFD+10] Karsten Schwenk, Tobias Alexander Franke, Timm Drevensek, Arjan Kuijper, Ulrich Bockholt, and Dieter Fellner. Adapting precomputed radiance transfer to real-time spectral rendering. In Hendrik Lensch, editor, *Eurographics*, Norrköping, Sweden, 2010

[JWO+10] Yvonne Jung, Sabine Webel, Manuel Olbrich, Timm Drevensek,

Tobias Alexander Franke, Markus Roth, and Dieter Fellner. Interactive textures as spatial user interfaces in x3d. In *Proceedings of the 15th International Conference on Web 3D Technology*, Web3D '10, pages 147–150, New York, NY, USA, 2010. ACM

2009

[JWKF09] Yvonne Jung, Christine Weber, Jens Keil, and Tobias Alexander Franke. Real-time rendering of skin changes caused by emotions. In Zsóia Ruttkay, Michael Kipp, Anton Nijholt, and HannesHögni Vilhjámsson, editors, *Intelligent Virtual Agents*, volume 5773 of *Lecture Notes in Computer Science*, pages 504–505. Springer Berlin Heidelberg, 2009

2008

[FJ08a] Tobias Alexander Franke and Yvonne Jung. Precomputed radiance transfer for x3d based mixed reality applications. In *Proceedings of the 13th international symposium on 3D web technology*, Web3D '08, pages 7–10, New York, NY, USA, 2008. ACM

[FJ08b] Tobias Alexander Franke and Yvonne Jung. Real-time mixed reality with gpu techniques. In *GRAPP 2008: Proceedings of the Third International Conference on Computer Vision Theory and Applications*, pages 249–252. INSTICC Press, 2008

2007

[JFDB07] Yvonne Jung, Tobias Alexander Franke, Patrick Dähne, and Johannes Behr. Enhancing x3d for advanced mr appliances. In *Proceedings of the Twelfth International Conference on 3D Web Technology*, Web3D '07, pages 27–36, New York, NY, USA, 2007. ACM

Journal Papers & Talks

2015

[Fra15] Tobias Alexander Franke. The state of the art in real-time relighting for augmented reality. ETH Zürich Visual Computing Talk, Jan 2015

2013

[LJB+13] Max Limper, Yvonne Jung, Johannes Behr, Timo Sturm, Tobias Alexander Franke, Karsten Schwenk, and Arjan Kuijper. Fast and progressive loading of binary encoded declarative 3d web content. *IEEE Computer Graphics and Applications*, 99(PrePrints):1, 2013

Bibliography

[Aal13] Frederik Aalund. A comparative study of screen-space ambient occlusion methods. Technical report, Technical University of Denmark, Building 321, DK-2800 Kongens Lyngby, Denmark, 2013.

[Ama84] John Amanatides. Ray tracing with cones. *SIGGRAPH Comput. Graph.*, 18(3):129–135, January 1984.

[AMH12] Ibrahim Arief, Simon McCallum, and Jon Yngve Hardeberg. Realtime estimation of illumination direction for augmented reality on mobile devices. In *20th Color and Imaging Conference, CIC 2012, Los Angeles, California, USA, November 12-16, 2012*, pages 111–116, 2012.

[ARBJ03] Sameer Agarwal, Ravi Ramamoorthi, Serge Belongie, and Henrik Wann Jensen. Structured importance sampling of environment maps. *ACM Trans. Graph.*, 22(3):605–612, July 2003.

[BCD+13] Francesco Banterle, Marco Callieri, Matteo Dellepiane, Massimiliano Corsini, Fabio Pellacini, and Roberto Scopigno. Envydepth: An interface for recovering local natural illumination from environment maps. *Computer Graphics Forum*, 32(7):411–420, October 2013. Proceedings of Pacific Graphics 2013.

[BGWK03] Oliver Bimber, Anselm Grundhöfer, Gordon Wetzstein, and Sebastian Knödel. Consistent illumination within optical see-through augmented environments. In *ACM SIGGRAPH 2003 Sketches &Amp; Applications*, SIGGRAPH '03, pages 1–1, New York, NY, USA, 2003. ACM.

[BJFS12] Johannes Behr, Yvonne Jung, Tobias Alexander Franke, and Timo Sturm. Using images and explicit binary container for efficient and incremental delivery of declarative 3d scenes on the web. In *Proceedings of the 17th International Conference on 3D Web Technology*, Web3D '12, pages 17–25, New York, NY, USA, 2012. ACM.

[Bla12] Victor Blacus. Electromagnetic spectrum. Wikimedia Commons, October 2012. CC BY-SA 3.0.

[BN76a] James F. Blinn and Martin E. Newell. Texture and reflection in computer generated images. *SIGGRAPH Comput. Graph.*, 10(2):266–266, July 1976.

[BN76b] James F. Blinn and Martin E. Newell. Texture and reflection in computer generated images. *Commun. ACM*, 19(10):542–547, October 1976.

[Bon14] Universität Bonn. Btfdbb: Btf database bonn and measurement lab, 2014.

[Bra00] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[BRD14] Stephan Bergmann, Tobias Ritschel, and Carsten Dachsbacher. Interactive Appearance Editing in RGB-D Images. In Jan Bender, Arjan Kuijper, Tatiana von Landesberger, Holger Theisel, and Philipp Urban, editors, *VMV 2014: Vision, Modeling & Visualization*, pages 1–8, Darmstadt, Germany, 2014. Eurographics Association.

[BS87] P. Beckmann and A. Spizzichino. *The Scattering of Electromagnetic Waves from Rough Surfaces*. Artech House, Inc., Norwood, MA, USA, 1987.

[BS09] Gabriele Bleser and Didier Stricker. Advanced tracking through efficient image processing and visual-inertial sensor fusion. *Computers & Graphics*, 33(1):59–72, 2009.

[BSD08] Louis Bavoil, Miguel Sainz, and Rouslan Dimitrov. Image-space horizon-based ambient occlusion. In *ACM SIGGRAPH 2008 Talks*, SIGGRAPH '08, pages 22:1–22:1, New York, NY, USA, 2008. ACM.

[BTS13] Markus Broecker, Bruce H. Thomas, and Ross T. Smith. Adapting ray tracing to spatial augmented reality. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 1–6, Oct 2013.

[CBP07] Luc Claustres, Loïc Barthe, and Mathias Paulin. Wavelet encoding of brdfs for real-time rendering. In *In Graphics Interface 07*, 2007.

[CDP+14] Guojun Chen, Yue Dong, Pieter Peers, Jiawan Zhang, and Xin Tong. Reflectance scanning: Estimating shading frame and brdf with generalized linear light sources. *ACM Transactions on Graphics*, 33(4), August 2014.

[CETC06] David Cline, Parris K Egbert, Justin F Talbot, and David L Cardon. Two stage importance sampling for direct lighting. In *Proceedings of the 17th Eurographics conference on Rendering Techniques*, pages 103–113. Eurographics Association, 2006.

[CJAMJ05] Petrik Clarberg, Wojciech Jarosz, Tomas Akenine-Möller, and Henrik Wann Jensen. Wavelet importance sampling: Efficiently evaluating products of complex functions. *ACM Trans. Graph.*, 24(3):1166–1175, July 2005.

[CJG12] Avishek Chatterjee, Suraj Jain, and Venu Madhav Govindu. A pipeline for building 3d models using depth cameras. In *Proceedings of the Eighth Indian Conference on Computer Vision, Graphics and Image Processing*, ICVGIP '12, pages 38:1–38:8, New York, NY, USA, 2012. ACM.

[CNR08] Oliver Cossairt, Shree Nayar, and Ravi Ramamoorthi. Light field transfer: Global illumination between real and synthetic objects. *ACM Trans. Graph.*, 27(3):57:1–57:6, August 2008.

[CNS+11] Cyril Crassin, Fabrice Neyret, Miguel Sainz, Simon Green, and Elmar Eisemann. Interactive indirect illumination using voxel cone tracing. *Computer Graphics Forum (Proceedings of Pacific Graphics 2011)*, 30(7), sep 2011.

[CT82] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. *ACM Trans. Graph.*, 1(1):7–24, January 1982.

[DBBS06] Philip Dutre, Kavita Bala, Philippe Bekaert, and Peter Shirley. *Advanced Global Illumination*. AK Peters Ltd, 2006.

[Deb98] Paul Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 189–198, New York, NY, USA, 1998. ACM.

[Deb05] Paul Debevec. A median cut algorithm for light probe sampling. In *ACM SIGGRAPH 2005 Posters*, SIGGRAPH '05, New York, NY, USA, 2005. ACM.

[DHT+00] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Tech-*

niques, SIGGRAPH '00, pages 145–156, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[DKH+14] Carsten Dachsbacher, Jaroslav Křivánek, Miloš Hašan, Adam Arbree, Bruce Walter, and Jan Novák. Scalable realistic rendering with many-light methods. *Computer Graphics Forum*, 33(1):88–104, 2014.

[DS05] Carsten Dachsbacher and Marc Stamminger. Reflective shadow maps. In *Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games*, I3D '05, pages 203–231, New York, NY, USA, 2005. ACM.

[DS06] Carsten Dachsbacher and Marc Stamminger. Splatting indirect illumination. In *Proceedings of the 2006 symposium on Interactive 3D graphics and games*, I3D '06, pages 93–100, New York, NY, USA, 2006. ACM.

[DSDD07] Carsten Dachsbacher, Marc Stamminger, George Drettakis, and Frédo Durand. Implicit visibility and antiradiance for interactive global illumination. *ACM Trans. Graph.*, 26(3), July 2007.

[dSS14] F. de Sorbier and H. Saito. Stereoscopic augmented reality with pseudo-realistic global illumination effects. In *Stereoscopic Displays and Applications XXV*, February 2014. accepted.

[DvGNK99] Kristin J. Dana, Bram van Ginneken, Shree K. Nayar, and Jan J. Koenderink. Reflectance and texture of real-world surfaces. *ACM Trans. Graph.*, 18(1):1–34, January 1999.

[FF12] Tobias Alexander Franke and Dieter W. Fellner. A scalable framework for image-based material representations. In *Proceedings of the 17th International Conference on 3D Web Technology*, Web3D '12, pages 83–91, New York, NY, USA, 2012. ACM.

[FG14] Simon Fuhrmann and Michael Goesele. Floating scale surface reconstruction. *ACM Trans. Graph.*, 33(4):46:1–46:11, July 2014.

[FGR92] Alain Fournier, Atjeng S. Gunawan, and Chris Romanzin. Common illumination between real and computer generated scenes. Technical report, University of British Columbia, Vancouver, BC, Canada, Canada, 1992.

[FH09] J. Filip and M. Haindl. Bidirectional texture function modeling: A state of the art survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(11):1921–1940, Nov 2009.

[FJ08a] Tobias Alexander Franke and Yvonne Jung. Precomputed radiance transfer for x3d based mixed reality applications. In *Proceedings of the 13th international symposium on 3D web technology*, Web3D '08, pages 7–10, New York, NY, USA, 2008. ACM.

[FJ08b] Tobias Alexander Franke and Yvonne Jung. Real-time mixed reality with gpu techniques. In *GRAPP 2008: Proceedings of the Third International Conference on Computer Vision Theory and Applications*, pages 249–252. INSTICC Press, 2008.

[FKOJ11] Tobias Alexander Franke, Svenja Kahn, Manuel Olbrich, and Yvonne Jung. Enhancing realism of mixed reality applications through real-time depth-imaging devices in x3d. In *Proceedings of the 16th International Conference on 3D Web Technology*, Web3D '11, pages 71–79, New York, NY, USA, 2011. ACM.

[FOF12] Tobias Alexander Franke, Manuel Olbrich, and Dieter W. Fellner. A flexible approach to gesture recognition and interaction in x3d. In *Proceedings of the 17th International Conference on 3D Web Technology*, Web3D '12, pages 171–174, New York, NY, USA, 2012. ACM.

[Fra13a] Tobias Alexander Franke. Delta light propagation volumes for mixed reality. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 125–132, Oct 2013.

[Fra13b] Tobias Alexander Franke. Near-field illumination for mixed reality with delta radiance fields. In *ACM SIGGRAPH 2013 Posters*, SIGGRAPH '13, pages 76:1–76:1, New York, NY, USA, 2013. ACM.

[Fra14a] Tobias Alexander Franke. Delta voxel cone tracing. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, pages 39–44, Sept 2014.

[Fra14b] Tobias Alexander Franke. Interactive relighting of arbitrary rough surfaces. In *ACM SIGGRAPH 2014 Posters*, SIGGRAPH '14, pages 28:1–28:1, New York, NY, USA, 2014. ACM.

[Fra15] Tobias Alexander Franke. The state of the art in real-time relighting for augmented reality. ETH Zürich Visual Computing Talk, Jan 2015.

[FSBR13] Tobias Alexander Franke, Volker Settgast, Johannes Behr, and Bruno Raffin. Vcore: a web resource oriented architecture for efficient data exchange. In *Proceedings of the 18th International Conference on 3D Web Technology*, Web3D '13, pages 71–78, New York, NY, USA, 2013. ACM.

[GCHH03] Simon Gibson, Jon Cook, Toby Howard, and Roger Hubbold. Rapid shadow generation in real-world lighting environments. In *Proceedings of the 14th Eurographics Workshop on Rendering*, EGRW '03, pages 219–229, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[GEM07] Thorsten Grosch, Tobias Eble, and Stefan Mueller. Consistent interactive augmentation of live camera images with correct

near-field illumination. In *Proceedings of the 2007 ACM symposium on Virtual reality software and technology*, VRST '07, pages 125–132, New York, NY, USA, 2007. ACM.

[GHH01] Simon Gibson, Toby Howard, and Roger Hubbold. Flexible image-based photometric reconstruction using virtual light sources. *Computer Graphics Forum*, 20:1067–7055, 2001.

[GKD07a] Paul Green, Jan Kautz, and Frédo Durand. Efficient reflectance and visibility approximations for environment map rendering. *Computer Graphics Forum (Proc. EUROGRAPHICS)*, 26(3):495–502, 2007.

[GKD07b] Paul Green, Jan Kautz, and Frédo Durand. Efficient reflectance and visibility approximations for environment map rendering. *Computer Graphics Forum*, 26(3):495–502, 2007.

[GLS+14] Lukas Gruber, Tobias Langlotz, Pradeep Sen, Tobias Hoherer, and Dieter Schmalstieg. Efficient and robust radiance transfer for probeless photorealistic augmented reality. In *2014 IEEE Virtual Reality, VR 2014, Minneapolis, MN, USA, March 29 - April 2, 2014*, pages 15–20, 2014.

[GM00] Simon Gibson and Alan Murta. Interactive rendering with real-world illumination. In *Proceedings of the Eurographics Workshop on Rendering Techniques 2000*, pages 365–376, London, UK, UK, 2000. Springer-Verlag.

[GMT39] Andreĭ Gershun, Parry Hiram Moon, and Gregory Timoshenko. *The light field*. Massachusetts Institute of Technology, 1939.

[Gou09] Brian Gough. *GNU Scientific Library Reference Manual - Third Edition*. Network Theory Ltd., 3rd edition, 2009.

[Gre03] Robin Green. Spherical Harmonic Lighting: The Gritty De-

tails. *Archives of the Game Developers Conference*, March 2003.

[Gro05] Thorsten Grosch. Differential photon mapping: Consistent augmentation of photographs with correction of all light paths. In M. Alexa and J. Marks, editors, *Eurographics 2005, EG Short Pres.*, pages 53–56, Trinity College, Dublin, Ireland, 2005. Eurographics Association.

[Gro07] Thorsten Grosch. *Augmentierte Bildsynthese*. PhD thesis, Universität Koblenz-Landau, 2007.

[GRTS12] L. Gruber, T. Richter-Trummer, and D. Schmalstieg. Real-time photometric registration from arbitrary geometry. In *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*, pages 119–128, Nov 2012.

[GSHG98] Gene Greger, Peter Shirley, Philip M. Hubbard, and Donald P. Greenberg. The irradiance volume. *IEEE Comput. Graph. Appl.*, 18(2):32–43, March 1998.

[GTGB84] Cindy M. Goral, Kenneth E. Torrance, Donald P. Greenberg, and Bennett Battaile. Modeling the interaction of light between diffuse surfaces. *SIGGRAPH Comput. Graph.*, 18(3):213–222, January 1984.

[HBR+11] Jing Huang, Tamy Boubekeur, Tobias Ritschel, Matthias Holländer, and Elmar Eisemann. Separable approximation of ambient occlusion. In *Eurographics 2011 - Short papers*, 2011.

[HDH03] Michael Haller, Stephan Drab, and Werner Hartmann. A real-time shadow approach for an augmented reality application using shadow volumes. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, VRST '03, pages 56–65, New York, NY, USA, 2003. ACM.

[HF14] Lukas Hermanns and Tobias Alexander Franke. Screen space

cone tracing for glossy reflections. In *ACM SIGGRAPH 2014 Posters*, SIGGRAPH '14, pages 102:1–102:1, New York, NY, USA, 2014. ACM.

[HFJS09] Timothy J. Hattenberger, Mark D. Fairchild, Garrett M. Johnson, and Carl Salvaggio. A psychophysical investigation of global illumination algorithms used in augmented reality. *ACM Trans. Appl. Percept.*, 6(1):2:1–2:22, February 2009.

[HKM11] Tomohito Hattori, Hiroyuki Kubo, and Shigeo Morishima. Real time ambient occlusion by curvature dependent occlusion function. In *SIGGRAPH Asia 2011 Posters*, SA '11, pages 48:1–48:1, New York, NY, USA, 2011. ACM.

[HMD+14] Stephen Hill, Stephen McAuley, Jonathan Dupuy, Yoshiharu Gotanda, Eric Heitz, Naty Hoffman, Sébastien Lagarde, Anders Langlands, Ian Megibben, Farhez Rayani, and Charles de Rousiers. Physically based shading in theory and practice. In *ACM SIGGRAPH 2014 Courses*, SIGGRAPH '14, pages 23:1–23:8, New York, NY, USA, 2014. ACM.

[HRKP04] Charles E Hughes, Erik Reinhard, Jaakko Konttinen, and Sumanta Pattanaik. Achieving interactive-time realistic illumination in mixed reality. Technical report, DTIC Document, 2004.

[IB98] Michael Isard and Andrew Blake. Condensation—conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

[IKH+11] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software*

and Technology, UIST '11, pages 559–568, New York, NY, USA, 2011. ACM.

[Jak10] Wenzel Jakob. Mitsuba renderer, 2010. http://www.mitsuba-renderer.org.

[JFDB07] Yvonne Jung, Tobias Alexander Franke, Patrick Dähne, and Johannes Behr. Enhancing x3d for advanced mr appliances. In *Proceedings of the Twelfth International Conference on 3D Web Technology*, Web3D '07, pages 27–36, New York, NY, USA, 2007. ACM.

[JWKF09] Yvonne Jung, Christine Weber, Jens Keil, and Tobias Alexander Franke. Real-time rendering of skin changes caused by emotions. In Zsóia Ruttkay, Michael Kipp, Anton Nijholt, and HannesHögni Vilhjámsson, editors, *Intelligent Virtual Agents*, volume 5773 of *Lecture Notes in Computer Science*, pages 504–505. Springer Berlin Heidelberg, 2009.

[JWO+10] Yvonne Jung, Sabine Webel, Manuel Olbrich, Timm Drevensek, Tobias Alexander Franke, Markus Roth, and Dieter Fellner. Interactive textures as spatial user interfaces in x3d. In *Proceedings of the 15th International Conference on Web 3D Technology*, Web3D '10, pages 147–150, New York, NY, USA, 2010. ACM.

[Kaj86] James T. Kajiya. The rendering equation. *SIGGRAPH Comput. Graph.*, 20(4):143–150, August 1986.

[KB99] Hirokazu Kato and Mark Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality*, IWAR '99, pages 85–94, Washington, DC, USA, 1999. IEEE Computer Society.

[KB12] Daniel Kurz and Selim Benhimane. Handheld augmented re-

ality involving gravity measurements. *Computers & Graphics*, 36(7):866–883, 2012.

[KC08] Jaroslav Křivánek and Mark Colbert. Real-time shading with filtered importance sampling. *Computer Graphics Forum*, 27(4):1147–1154, 2008.

[KD10] Anton Kaplanyan and Carsten Dachsbacher. Cascaded light propagation volumes for real-time indirect illumination. In *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '10, pages 99–107, New York, NY, USA, 2010. ACM.

[Kel97] Alexander Keller. Instant radiosity. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, pages 49–56, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[KF14] Kevin Karsch and David Forsyth. Blind recovery of spatially varying reflectance from a single image. In *SIGGRAPH Asia Workshop on Indoor Scene Understanding: Where Graphics meets Vision*, 2014.

[KFB10] Jaroslav Křivánek, James A. Ferwerda, and Kavita Bala. Effects of global illumination approximations on material appearance. *ACM Trans. Graph.*, 29(4):112:1–112:10, July 2010.

[KHFH11] Kevin Karsch, Varsha Hedau, David Forsyth, and Derek Hoiem. Rendering synthetic objects into legacy photographs. *ACM Trans. Graph.*, 30(6):157:1–157:12, December 2011.

[KK03] Thomas Kollig and Alexander Keller. Efficient illumination by high dynamic range images. In *Rendering Techniques*, pages 45–51, 2003.

[KK07] Oscar Kozlowski and Jan Kautz. Is accurate occlusion of glossy reflections necessary? In *Proceedings of the 4th Sym-*

posium on Applied Perception in Graphics and Visualization, APGV '07, pages 91–98, New York, NY, USA, 2007. ACM.

[KK10] Peter Kán and Hannes Kaufmann. High-quality real-time global illumination in augmented reality, January 2010. https://www.ims.tuwien.ac.at/projects/rayengine.

[KK12] Peter Kán and Hannes Kaufmann. High-quality reflections, refractions, and caustics in augmented reality and their contribution to visual coherence. In *Proceedings of the 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, ISMAR '12, pages 99–108, Washington, DC, USA, 2012. IEEE Computer Society.

[KK13] Peter Kán and Hannes Kaufmann. Differential irradiance caching for fast high-quality light transport between virtual and real worlds. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 133–141, Oct 2013.

[KK14] Sebastian B. Knorr and Daniel Kurz. Real-time illumination estimation from faces for coherent rendering. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, pages 113–122, Sept 2014.

[KKG+14] Jaroslav Křivánek, Alexander Keller, Iliyan Georgiev, Anton S. Kaplanyan, Marcos Fajardo, Mark Meyer, Jean-Daniel Nahmias, Ondřej Karlík, and Juan Cañada. Recent advances in light transport simulation: Some theory and a lot of practice. In *ACM SIGGRAPH 2014 Courses*, SIGGRAPH '14, pages 17:1–17:6, New York, NY, USA, 2014. ACM.

[KKT11] P. Kuhtreiber, M. Knecht, and C. Traxler. Brdf approximation and estimation for augmented reality. In *System Theory, Control, and Computing (ICSTCC), 2011 15th International Conference on*, pages 1–6, Oct 2011.

[KLK14] Kevin Karsch, Ce Liu, and Sing Bing Kang. Depthtransfer: Depth extraction from video using non-parametric sampling. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2014.

[KM08] Georg Klein and David Murray. Compositing for small cameras. In *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, ISMAR '08, pages 57–60, Washington, DC, USA, 2008. IEEE Computer Society.

[KM10] Georg Klein and David W. Murray. Simulating low-cost cameras for augmented reality compositing. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):369–380, 2010.

[KOI05] T. Kakuta, T. Oishi, and K. Ikeuchi. Virtual kawaradera: Fast shadow texture for augmented reality. In *CREST05*, pages 79–85, 2005.

[KSA13] Viktor Kämpe, Erik Sintorn, and Ulf Assarsson. High resolution sparse voxel dags. *ACM Trans. Graph.*, 32(4):101:1–101:13, July 2013.

[KSF10] E. Kruijff, J.E. Swan, and S. Feiner. Perceptual issues in augmented reality revisited. In *Mixed and Augmented Reality (ISMAR), 2010 9th IEEE International Symposium on*, pages 3–12, Oct 2010.

[KSH+14] K Karsch, K. Sunkavalli, S. Hadap, N. Carr, H. Jin, R. Fonte, M. Sittig, and D. Forsyth. Automatic scene inference for 3d object compositing. *ACM Trans. Graph.*, 33(3), June 2014.

[KSS+04] Jan Kautz, Mirko Sattler, Ralf Sarlette, Reinhard Klein, and Hans-Peter Seidel. Decoupling brdfs from surface mesostructures. In *Proceedings of Graphics Interface 2004*, GI '04, pages 177–182, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2004. Canadian Human-Computer Communications Society.

[KTM+10] Martin Knecht, Christoph Traxler, Oliver Mattausch, Werner Purgathofer, and Michael Wimmer. Differential Instant Radiosity for Mixed Reality. In *Proceedings of the 2010 IEEE International Symposium on Mixed and Augmented Reality (ISMAR 2010)*, pages 99–107, October 2010.

[KTPW11] Martin Knecht, Christoph Traxler, Werner Purgathofer, and Michael Wimmer. Adaptive camera-based color mapping for mixed-reality applications. In *Proceedings of the 2011 IEEE International Symposium on Mixed and Augmented Reality (ISMAR 2011)*, pages 165–168. IEEE/IET Electronic Library (IEL), IEEE-Wiley eBooks Library, VDE VERLAG Conference Proceedings, October 2011. E-ISBN: 978-1-4577-2184-7.

[KTTW12] Martin Knecht, Georg Tanzmeister, Christoph Traxler, and Michael Wimmer. Interactive brdf estimation for mixed-reality applications. *Journal of WSCG*, 20(1):47–56, June 2012.

[KTWW13] M. Knecht, C. Traxler, C. Winklhofer, and M. Wimmer. Reflective and refractive objects for mixed reality. *IEEE Transactions on Visualization and Computer Graphics*, 19(4):576–582, 2013.

[Lab06] Mitsubishi Electric Research Laboratories. Merl brdf database, 2006.

[Lan02] Hayden Landis. Production-ready global illumination. *Siggraph course notes*, 16(2002):11, 2002.

[LB12] Philipp Lensing and Wolfgang Broll. Instant indirect illumination for dynamic mixed reality scenes. In *Proceedings of the 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, ISMAR '12, pages 109–118, Washington, DC, USA, 2012. IEEE Computer Society.

[LB13a] Philipp Lensing and Wolfgang Broll. Efficient shading of in-

direct illumination applying reflective shadow maps. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '13, pages 95–102, New York, NY, USA, 2013. ACM.

[LB13b] Philipp Lensing and Wolfgang Broll. Lightskin: Real-time global illumination for virtual and mixed reality. In *Joint Virtual Reality Conference of EGVE - EuroVR, Paris, France, 2013. Proceedings*, pages 17–24, 2013.

[LB13c] Philipp Lensing and Wolfgang Broll. LightSkin: Real-Time Global Illumination for Virtual and Mixed Reality. In *Joint Virtual Reality Conference of EGVE - EuroVR*, pages 17–24, Paris, France, 2013. Eurographics Association.

[LCD06] Thomas Luft, Carsten Colditz, and Oliver Deussen. Image enhancement by unsharp masking the depth buffer. *ACM Trans. Graph.*, 25(3):1206–1213, July 2006.

[LEN12] Jean-François Lalonde, AlexeiA. Efros, and SrinivasaG. Narasimhan. Estimating the natural illumination conditions from a single outdoor image. *International Journal of Computer Vision*, 98(2):123–145, 2012.

[LJB+13] Max Limper, Yvonne Jung, Johannes Behr, Timo Sturm, Tobias Alexander Franke, Karsten Schwenk, and Arjan Kuijper. Fast and progressive loading of binary encoded declarative 3d web content. *IEEE Computer Graphics and Applications*, 99(PrePrints):1, 2013.

[LK81] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[LK03] Jaakko Lehtinen and Jan Kautz. Matrix radiance transfer. In

Proceedings of the 2003 Symposium on Interactive 3D Graphics, I3D '03, pages 59–64, New York, NY, USA, 2003. ACM.

[LKG+03] Hendrik P. A. Lensch, Jan Kautz, Michael Goesele, Wolfgang Heidrich, and Hans-Peter Seidel. Image-based reconstruction of spatial appearance and geometric detail. *ACM Trans. Graph.*, 22(2):234–257, April 2003.

[LMHRG10] Jorge Lopez-Moreno, Sunil Hadap, Erik Reinhard, and Diego Gutierrez. Compositing images through light source detection. *Computers & Graphics*, 34(6):698 – 707, 2010. Graphics for Serious Games Computer Graphics in Spain: a Selection of Papers from {CEIG} 2009 Selected Papers from the {SIGGRAPH} Asia Education Program.

[LNJS12] Bradford J. Loos, Derek Nowrouzezahrai, Wojciech Jarosz, and Peter-Pike Sloan. Delta radiance transfer. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '12, pages 191–196, New York, NY, USA, 2012. ACM.

[LQX+09] Yanli Liu, Xueying Qin, Songhua Xu, Eihachiro Nakamae, and Qunsheng Peng. Light source estimation of outdoor scenes for mixed reality. *The Visual Computer*, 25(5-7):637–646, 2009.

[LSK+07] Samuli Laine, Hannu Saransaari, Janne Kontkanen, Jaakko Lehtinen, and Timo Aila. Incremental instant radiosity for real-time indirect illumination. In *Proceedings of the 18th Eurographics Conference on Rendering Techniques*, EGSR'07, pages 277–286, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[Mad03] ClausB. Madsen. Using real shadows to create virtual ones. In Josef Bigun and Tomas Gustavsson, editors, *Image Analysis*, volume 2749 of *Lecture Notes in Computer Science*, pages 820–827. Springer Berlin Heidelberg, 2003.

[Mad08] Claus B. Madsen. Estimating radiances of the sun and the sky from a single image containing shadows. In *Proceedings: 16th Danish Conference on Pattern Recognition and Image Analysis*, Koebenhavns Universitet. Datalogisk Institut. Rapport. Department of Computer Science, University of Copenhagen, 2008.

[Mar69] Fulgence Marion. *The wonders of optics.* Wonders of science. C. Scribner & co., New York, 1869. http://archive.org/details/wondersofoptics00mariiala.

[Mar98] Stephen Robert Marschner. *Inverse Rendering for Computer Graphics.* PhD thesis, Cornell University, Ithaca, NY, USA, 1998. AAI9839924.

[MGW01] Tom Malzbender, Dan Gelb, and Hans Wolters. Polynomial texture maps. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 519–528, New York, NY, USA, 2001. ACM.

[MH84] G. S. Miller and C. R. Hoffman. Illumination and reflection maps: Simulated objects in simulated and real environments. In *SIGGRAPH 84: Advanced Computer Graphics Animation Seminar Note*, SIGGRAPH '84. ACM, July 1984.

[Mit07] Martin Mittring. Finding next gen: Cryengine 2. In *ACM SIGGRAPH 2007 Courses*, SIGGRAPH '07, pages 97–121, New York, NY, USA, 2007. ACM.

[ML10] Claus B. Madsen and Brajesh Behari Lal. Probeless illumination estimation for outdoor augmented reality. In *Augmented Reality.* INTECH, 2010.

[ML13] ClausB. Madsen and BrajeshB. Lal. Estimating outdoor illumination conditions based on detection of dynamic shadows. In Gabriela Csurka, Martin Kraus, Leonid Mestetskiy,

Paul Richard, and José Braz, editors, *Computer Vision, Imaging and Computer Graphics. Theory and Applications*, volume 274 of *Communications in Computer and Information Science*, pages 33–52. Springer Berlin Heidelberg, 2013.

[MMK03] Gero Müller, Jan Meseth, and Reinhard Klein. Compression and real-time rendering of measured btfs using local pca. In T. Ertl, B. Girod, G. Greiner, H. Niemann, H.-P. Seidel, E. Steinbach, and R. Westermann, editors, *Vision, Modeling and Visualisation 2003*, pages 271–280. Akademische Verlagsgesellschaft Aka GmbH, Berlin, November 2003.

[MML12] Morgan McGuire, Michael Mara, and David Luebke. Scalable ambient obscurance. In *High-Performance Graphics 2012*, June 2012.

[MMNL14] Michael Mara, Morgan McGuire, Derek Nowrouzezahrai, and David Luebke. Fast global illumination approximations on deep g-buffers. Technical Report NVR-2014-001, NVIDIA Corporation, June 2014.

[MMS+05] Gero Müller, Jan Meseth, Mirko Sattler, Ralf Sarlette, and Reinhard Klein. Acquisition, synthesis and rendering of bidirectional texture functions. *Computer Graphics Forum*, 24(1):83–109, March 2005.

[MN08] Claus B. Madsen and Michael Nielsen. Towards probe-less augmented reality - A position paper. In *Real-Time Mixed Reality with GPU Techniques.*, pages 255–261, 2008.

[MOBH11] Morgan McGuire, Brian Osman, Michael Bukowski, and Padraic Hennessy. The alchemy screen-space ambient obscurance algorithm. In *High-Performance Graphics 2011*, August 2011.

[MPBM03] Wojciech Matusik, Hanspeter Pfister, Matt Brand, and

Leonard McMillan. A data-driven reflectance model. *ACM Trans. Graph.*, 22(3):759–769, July 2003.

[MSW10] Oliver Mattausch, Daniel Scherzer, and Michael Wimmer. High-quality screen-space ambient occlusion using temporal coherence. *Computer Graphics Forum*, 29(8), December 2010.

[MTUK95] Paul Milgram, Haruo Takemura, Akira Utsumi, and Fumio Kishino. Augmented reality: a class of displays on the reality-virtuality continuum. In *Proceedings of the SPIE*, volume 2351, pages 282–292. International Society for Optics and Photonics, 1995.

[Mül09] Gero Müller. *Data-Driven Methods for Compression and Editing of Spatially Varying Appearance*. Dissertation, Universität Bonn, December 2009.

[MWLT00] Stephen R. Marschner, Stephen H. Westin, Eric P. F. Lafortune, and Kenneth E. Torrance. Image-based bidirectional reflectance distribution function measurement. *Appl. Opt.*, 39(16):2592–2600, Jun 2000.

[NED11] Jan Novák, Thomas Engelhardt, and Carsten Dachsbacher. Screen-space bias compensation for interactive high-quality global illumination with virtual point lights. In *Symposium on Interactive 3D Graphics and Games*, I3D '11, pages 119–124, New York, NY, USA, 2011. ACM.

[NGM+11] Derek Nowrouzezahrai, Stefan Geiger, Kenny Mitchell, Robert Sumner, Wojciech Jarosz, and Markus Gross. Light factorization for mixed-frequency shadows in augmented reality. In *Proceedings of the 2011 IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '11, pages 173–179, Washington, DC, USA, 2011. IEEE Computer Society.

[NHIN86] Eihachiro Nakamae, Koichi Harada, Takao Ishizaki, and Tomoyuki Nishita. A montage method: The overlaying of the

computer generated images onto a background photograph. *SIGGRAPH Comput. Graph.*, 20(4):207–214, August 1986.

[NM07a] Michael Nielsen and ClausB. Madsen. Graph cut based segmentation of soft shadows for seamless removal and augmentation. In BjarneKjær Ersbøll and KimSteenstrup Pedersen, editors, *Image Analysis*, volume 4522 of *Lecture Notes in Computer Science*, pages 918–927. Springer Berlin Heidelberg, 2007.

[NM07b] Michael Nielsen and ClausB Madsen. Segmentation of soft shadows based on a daylight- and penumbra model. In André Gagalowicz and Wilfried Philips, editors, *Computer Vision/Computer Graphics Collaboration Techniques*, volume 4418 of *Lecture Notes in Computer Science*, pages 341–352. Springer Berlin Heidelberg, 2007.

[NRH04] Ren Ng, Ravi Ramamoorthi, and Pat Hanrahan. Triple product wavelet integrals for all-frequency relighting. *ACM Trans. Graph.*, 23(3):477–487, August 2004.

[NRS14] Oliver Nalbach, Tobias Ritschel, and Hans-Peter Seidel. Deep screen space. In *Proceedings of the 18th Meeting of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '14, pages 79–86, New York, NY, USA, 2014. ACM.

[NW09] Greg Nichols and Chris Wyman. Multiresolution splatting for indirect illumination. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games*, I3D '09, pages 83–90, New York, NY, USA, 2009. ACM.

[OA11] Ola Olsson and Ulf Assarsson. Tiled shading. *Journal of Graphics, GPU, and Game Tools*, 15(4):235–251, 2011.

[ODJ04] Victor Ostromoukhov, Charles Donohue, and Pierre-Marc Jodoin. Fast hierarchical importance sampling with blue noise

properties. *ACM Transactions on Graphics*, 23(3):488–495, 2004. Proc. SIGGRAPH 2004.

[OFKH13] Manuel Olbrich, Tobias Alexander Franke, Jens Keil, and Sven Hertling. Skeletal input for user interaction in x3d. In *Proceedings of the 18th International Conference on 3D Web Technology*, Web3D '13, pages 143–146, New York, NY, USA, 2013. ACM.

[OFR14] Manuel Olbrich, Tobias Alexander Franke, and Pavel Rojtberg. Remote visual tracking for the (mobile) web. In *Proceedings of the Nineteenth International ACM Conference on 3D Web Technologies*, Web3D '14, pages 27–33, New York, NY, USA, 2014. ACM.

[OHSG12] T. Oskam, A. Hornung, R.W. Sumner, and M. Gross. Fast and stable color balancing for images and augmented reality. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, pages 49–56, Oct 2012.

[ON94] Michael Oren and Shree K. Nayar. Generalization of lambert's reflectance model. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, pages 239–246, New York, NY, USA, 1994. ACM.

[Pap11] Georgios Papaioannou. Real-time diffuse global illumination using radiance hints. In *Proceedings of the ACM SIGGRAPH Symposium on High Performance Graphics*, HPG '11, pages 15–24, New York, NY, USA, 2011. ACM.

[PH04] Matt Pharr and Greg Humphreys. *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[PHD06] Pieter Peers, Tim Hawkins, and Paul Debevec. A reflective

light stage. Technical Report ICT-TR-04.2006, Institute for Creative Technologies, USC, December 2006.

[PKD12] Roman Prutkin, Anton Kaplanyan, and Carsten Dachsbacher. Reflective shadow map clustering for real-time global illumination. In *Eurographics (Short Papers)*, pages 9–12, 2012.

[PLW12] Youngmin Park, V. Lepetit, and Woontack Woo. Handling motion-blur in 3d tracking and rendering for augmented reality. *Visualization and Computer Graphics, IEEE Transactions on*, 18(9):1449–1459, Sept 2012.

[PP03] Gustavo Patow and Xavier Pueyo. A survey of inverse rendering problems. *Computer Graphics Forum*, 22(4):663–687, 2003.

[PTVF07] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 3 edition, 2007.

[PWL+07] Minghao Pan, Rui Wang, Xinguo Liu, Qunsheng Peng, and Hujun Bao. Precomputed radiance transfer field for rendering interreflections in dynamic scenes. *Comput. Graph. Forum*, 26(3):485–493, 2007.

[PWXLPB07] Minghao Pan, Rui Wang Xinguo Liu, Qunsheng Peng, and Hujun Bao. Precomputed radiance transfer field for rendering interreflections in dynamic scenes. *Computer Graphics Forum*, 26(3):485–493, 2007.

[RBDG14] Kai Rohmer, Wolfgang Buschel, Raimund Dachselt, and Thorsten Grosch. Interactive near-field illumination for photorealistic augmented reality on mobile devices. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, pages 29–38, Sept 2014.

[RBF08] Ganesh Ramanarayanan, Kavita Bala, and James A. Ferwerda. Perception of complex aggregates. *ACM Trans. Graph.*, 27(3):60:1–60:10, August 2008.

[RDGK12] Tobias Ritschel, Carsten Dachsbacher, Thorsten Grosch, and Jan Kautz. The state of the art in interactive global illumination. *Comput. Graph. Forum*, 31(1):160–188, February 2012.

[RFWB07] Ganesh Ramanarayanan, James Ferwerda, Bruce Walter, and Kavita Bala. Visual equivalence: Towards a new standard for image fidelity. *ACM Trans. Graph.*, 26(3), July 2007.

[RGK+08] T. Ritschel, T. Grosch, M. H. Kim, H.-P. Seidel, C. Dachsbacher, and J. Kautz. Imperfect shadow maps for efficient computation of indirect illumination. *ACM Trans. Graph.*, 27(5):129:1–129:8, December 2008.

[RGS09] Tobias Ritschel, Thorsten Grosch, and Hans-Peter Seidel. Approximating dynamic global illumination in image space. In *Proceedings of the 2009 Symposium on Interactive 3D Graphics and Games*, I3D '09, pages 75–82, New York, NY, USA, 2009. ACM.

[RH01] Ravi Ramamoorthi and Pat Hanrahan. An efficient representation for irradiance environment maps. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 497–500, New York, NY, USA, 2001. ACM.

[Rob31] E.G. Robertson. *Mémoires: récréatifs, scientifiques et anecdotiques d'un physicien-aéronaute E.G. Robertson*, volume 1. Chez l'auteur et á la Librairie de Wurtz, Paris, 1831. https://archive.org/details/mmoiresrcratifss01robe.

[RSK09] Martin Rump, Ralf Sarlette, and Reinhard Klein. Efficient resampling, compression and rendering of metallic and pearlescent paint. In M. Magnor, B. Rosenhahn, and H. Theisel,

editors, *Vision, Modeling, and Visualization*, pages 11–18, November 2009.

[SCCN11] Yu Sheng, Barbara Cutler, Chao Chen, and Joshua Nasman. Perceptual global illumination cancellation in complex projection environments. In *Proceedings of the Twenty-second Eurographics Conference on Rendering*, EGSR '11, pages 1261–1268, Aire-la-Ville, Switzerland, Switzerland, 2011. Eurographics Association.

[Sch94] Christophe Schlick. An inexpensive brdf model for physically-based rendering. *Computer Graphics Forum*, 13:233–246, 1994.

[SFD+10] Karsten Schwenk, Tobias Alexander Franke, Timm Drevensek, Arjan Kuijper, Ulrich Bockholt, and Dieter Fellner. Adapting precomputed radiance transfer to real-time spectral rendering. In Hendrik Lensch, editor, *Eurographics*, Norrköping, Sweden, 2010.

[SG69] J. Spanier and E.M. Gelbard. *Monte Carlo Principles and Neutron Transport Problems*. Addison-Wesley series in computer science and information processing. Addison-Wesley, 1969.

[SJS14] William Steptoe, Simon Julier, and Anthony Steed. Presence and discernability in conventional and non-photorealistic immersive augmented reality. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*, pages 213–218, Sept 2014.

[SKS02] Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph.*, 21(3):527–536, July 2002.

[SS10] Michael Schwarz and Hans-Peter Seidel. Fast parallel sur-

face and solid voxelization on gpus. *ACM Trans. Graph.*, 29(6):179:1–179:10, 2010.

[SSI99] I Sato, Y. Sato, and K. Ikeuchi. Illumination distribution from brightness in shadows: Adaptive estimation of illumination distribution with unknown reflectance properties in shadow regions. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 875–882 vol.2, 1999.

[SSI03] I. Sato, Y. Sato, and K. Ikeuchi. Illumination from shadows. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(3):290–300, March 2003.

[SSK03] Mirko Sattler, Ralf Sarlette, and Reinhard Klein. Efficient and realistic visualization of cloth. In *Eurographics Symposium on Rendering 2003*, June 2003.

[SSW+14] Christopher Schwartz, Ralf Sarlette, Michael Weinmann, Martin Rump, and Reinhard Klein. Design and implementation of practical bidirectional texture function measurement devices focusing on the developments at the university of bonn. *Sensors*, 14(5), April 2014.

[SSWK13] Christopher Schwartz, Ralf Sarlette, Michael Weinmann, and Reinhard Klein. Dome ii: A parallelized btf acquisition system. In Holly Rushmeier and Reinhard Klein, editors, *Eurographics Workshop on Material Appearance Modeling: Issues and Acquisition*, pages 25–31. Eurographics Association, June 2013.

[SWR+11] Christopher Schwartz, Michael Weinmann, Roland Ruiters, Arno Zinke, Ralf Sarlette, and Reinhard Klein. Capturing shape and reflectance of food. In *SIGGRAPH Asia 2011 Sketches*, SA '11, pages 28:1–28:2, New York, NY, USA, December 2011. ACM.

[SWRK11] Christopher Schwartz, Michael Weinmann, Roland Ruiters, and Reinhard Klein. Integrated high-quality acquisition of geometry and appearance for cultural heritage. In *The 12th International Symposium on Virtual Reality, Archeology and Cultural Heritage VAST 2011*, pages 25–32. Eurographics Association, Eurographics Association, October 2011.

[SYC10] Yu Sheng, Theodore C. Yapo, and Barbara Cutler. Global illumination compensation for spatially augmented reality. *Comput. Graph. Forum*, 29(2):387–396, 2010.

[SZ12] Lagarde Sébastien and Antoine Zanuttini. Local image-based lighting with parallax-corrected cubemaps. In *ACM SIGGRAPH 2012 Talks*, SIGGRAPH '12, pages 36:1–36:1, New York, NY, USA, 2012. ACM.

[TFG+13] Borom Tunwattanapong, Graham Fyffe, Paul Graham, Jay Busch, Xueming Yu, Abhijeet Ghosh, and Paul Debevec. Acquiring reflectance and shape from continuous spherical harmonic illumination. *ACM Trans. Graph.*, 32(4):109:1–109:12, July 2013.

[Tim13] Ville Timonen. Screen-Space Far-Field Ambient Obscurance. In *Proceedings of HPG 2013*, pages 33–43. ACM, 2013.

[TKD+14] Natasha Tatarchuk, Brian Karis, Michal Drobot, Nicolas Schulz, Jerome Charles, and Theodor Mader. Advances in real-time rendering in games, part i. In *ACM SIGGRAPH 2014 Courses*, SIGGRAPH '14, pages 10:1–10:1, New York, NY, USA, 2014. ACM.

[TL04] Eric Tabellion and Arnauld Lamorlette. An approximate global illumination system for computer generated films. *ACM Trans. Graph.*, 23(3):469–476, August 2004.

[TR75] T. S. Trowbridge and K. P. Reitz. Average irregularity rep-

resentation of a rough surface for ray reflection. *J. Opt. Soc. Am.*, 65(5):531–536, May 1975.

[Tra09] Christoph Traxler. Reciprocal shading for mixed reality, February 2009. http://www.cg.tuwien.ac.at/research/projects/RESHADE/.

[TS67] K. E. Torrance and E. M. Sparrow. Theory for off-specular reflection from roughened surfaces. *J. Opt. Soc. Am.*, 57(9):1105–1112, Sep 1967.

[TVB+14] Natasha Tatarchuk, Michal Valient, Wade Brainerd, Bartlomiej Wronski, Peter Sikachev, and Jean-Normand Bucci. Advances in real-time rendering in games, part ii. In *ACM SIGGRAPH 2014 Courses*, SIGGRAPH '14, pages 11:1–11:1, New York, NY, USA, 2014. ACM.

[VD09] Kuntee Viriyothai and Paul Debevec. Variance minimization light probe sampling. In *SIGGRAPH '09: Posters*, SIGGRAPH '09, pages 92:1–92:1, New York, NY, USA, 2009. ACM.

[Vea98] Eric Veach. *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, Stanford, CA, USA, 1998. AAI9837162.

[vK70] Wilhelm von Kügelgen. *Jugenderinnerungen eines alten Mannes*. Hertz, Berlin, 1870. http://reader.digitale-sammlungen.de/de/fs1/object/display/bsb10702022_00007.html.

[Wal12] Walt Disney Animation Studios. Brdf explorer, 2012.

[War92] Gregory J. Ward. Measuring and modeling anisotropic reflection. *SIGGRAPH Comput. Graph.*, 26(2):265–272, July 1992.

[WAT92] Stephen H. Westin, James R. Arvo, and Kenneth E. Torrance.

Predicting reflectance functions from complex surfaces. *SIGGRAPH Comput. Graph.*, 26(2):255–264, July 1992.

[WFA+05] Bruce Walter, Sebastian Fernandez, Adam Arbree, Kavita Bala, Michael Donikian, and Donald P. Greenberg. Lightcuts: A scalable approach to illumination. *ACM Trans. Graph.*, 24(3):1098–1107, July 2005.

[WFKO13] Sabine Webel, Tobias Alexander Franke, Jens Keil, and Manuel Olbrich. Immersive experience of current and ancient reconstructed cultural attractions . In *Digital Heritage International Congress - Track 2 - - Visualization & Interaction*, pages 395–398. Eurographics Association, 2013.

[Wil83] Lance Williams. Pyramidal parametrics. *SIGGRAPH Comput. Graph.*, 17(3):1–11, July 1983.

[WMLT07] Bruce Walter, Stephen R. Marschner, Hongsong Li, and Kenneth E. Torrance. Microfacet models for refraction through rough surfaces. In *Proceedings of the 18th Eurographics Conference on Rendering Techniques*, EGSR'07, pages 195–206, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[WRC07] Gregory J. Ward, Francis M. Rubinstein, and Robert D. Clear. A ray tracing solution for diffuse interreflection. In *ACM SIGGRAPH 2007 Courses*, SIGGRAPH '07, New York, NY, USA, 2007. ACM.

[WRD08] Greg Ward, Erik Reinhard, and Paul Debevec. High dynamic range imaging & image-based lighting. In *ACM SIGGRAPH 2008 Classes*, SIGGRAPH '08, pages 27:1–27:137, New York, NY, USA, 2008. ACM.

[Wue08] Harald Wuest. *Efficient Line and Patch Feature Characterization and Management for Real-time Camera Tracking*. PhD thesis, TU Darmstadt, November 2008.

[WVS05] H. Wuest, F. Vial, and D. Strieker. Adaptive line tracking with multiple hypotheses for augmented reality. In *Mixed and Augmented Reality, 2005. Proceedings. Fourth IEEE and ACM International Symposium on*, pages 62–69, Oct 2005.

[WZT+08] Jiaping Wang, Shuang Zhao, Xin Tong, John Snyder, and Baining Guo. Modeling anisotropic surface reflectance with example-based microfacet synthesis. *ACM Trans. Graph.*, 27(3):41:1–41:9, August 2008.

[XZL+13] GuanYu Xing, XueHong Zhou, YanLi Liu, XueYing Qin, and QunSheng Peng. Online illumination estimation of outdoor scenes based on videos containing no shadow area. *Science China Information Sciences*, 56(3):1–11, 2013.

[YCK+09] Insu Yu, Andrew Cox, Min H. Kim, Tobias Ritschel, Thorsten Grosch, Carsten Dachsbacher, and Jan Kautz. Perceptual influence of approximate visibility in indirect illumination. *ACM Trans. Appl. Percept.*, 6(4):24:1–24:14, October 2009.

[YDMH99] Yizhou Yu, Paul Debevec, Jitendra Malik, and Tim Hawkins. Inverse global illumination: Recovering reflectance models of real scenes from photographs. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 215–224, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[ZHL+05] Kun Zhou, Yaohua Hu, Stephen Lin, Baining Guo, and Heung-Yeung Shum. Precomputed shadow fields for dynamic scenes. *ACM Trans. Graph.*, 24(3):1196–1201, July 2005.